DIMAZ ANKAA WIJAYA

BITCOIN
TINGKAT LANJUT



puspantara

Bitcoin Tingkat Lanjut

DIMAZ ANKAA WIJAYA



Bitcoin Tingkat Lanjut

Diterbitkan pertama kali oleh Puspantara
Jalan Abadi Komplek Abadi Place Blok E-01,
Tanjung Rejo, Medan, Sumatera Utara, Indonesia
info@puspantara.org | puspantara.org

Editor: Fauzan Nur Ahmadi, Vinsensius Sitepu
Desainer grafis: Muhammad Zahrawi

ISBN: 978-602-60266-1-3



PENGANTAR PENULIS

Saat Bhagwan Chowdhry, pakar ekonomi dan akademisi dari UCLA, menominasikan Satoshi Nakamoto sebagai kandidat penerima Nobel untuk sains ekonomi tahun 2016, ia tidak sedang bergurau. Bitcoin, sistem yang diperkenalkannya ke publik di tahun 2009 lalu, telah menyita banyak perhatian. Bitcoin tidak hanya mempopulerkan kembali sistem peer-to-peer, melainkan juga telah menggugah pemikiran para intelektual untuk mengeksplorasi segala sisi Bitcoin. Bitcoin hanyalah permulaan. Kita akan melihat revolusi teknologi besar-besaran di masa depan dengan ide yang diperkenalkan Nakamoto.

Semua orang yang tertarik dengan cryptocurrency dan hendak mempelajarinya lebih lanjut sangat tepat jika bertitik tolak dari Bitcoin. Bitcoin memiliki hal-hal teknis yang menarik dipelajari, seperti yang disajikan oleh buku ini. Buku ini membahas beberapa hal yang ada dalam Bitcoin dengan lebih mendetail.

Bagian Pertama membahas landasan teori tentang Bitcoin: apa dan bagaimana sistem Bitcoin berjalan, kelebihan, dan kekurangan Bitcoin.

Bagian Kedua berisi tentang beberapa teknik kriptografi dan teknologi yang diimplementasikan di dalam Bitcoin.

Bagian Ketiga khusus mengulas tentang privasi dan anonimitas yang terkait dengan Bitcoin. Bagian ini juga menyebutkan analisis transaksi Bitcoin dan solusi-solusi anonimitas yang tersedia.

Bagian Keempat didesain bagi Anda yang tertarik untuk memahami lebih jauh tentang fitur scripting yang disematkan di dalam sistem Bitcoin.

Setelah membaca buku ini, Anda diharapkan dapat memahami mengapa banyak orang jatuh cinta pada Bitcoin dan cryptocurrency. Di era Internet seperti sekarang, tipe mata uang berbasis kriptografi ini tampaknya cukup menjanjikan untuk diimplementasikan.

Anda juga mungkin akan tertarik dengan solusi-solusi berbasis blockchain lainnya seperti smart contract yang diaplikasikan pada berbagai bidang teknologi lain seperti Internet of Things dan keamanan sistem.

Penulis berterima kasih kepada seluruh anggota tim Forensik Digital Direktorat Jenderal Pajak: Pak Andri, Izazi, Dony, Fatwa, Wishnu, Budi, dan teman-teman lain yang tidak dapat disebutkan satu-persatu. Penulis juga memberikan apresiasi setinggi-tingginya Lembaga Pengelola Dana Pendidikan, Kementerian Keuangan. Penulis juga tidak lupa mengucapkan terima kasih kepada para supervisor yang masih saja berperan penting, bahkan ketika penulis telah menyelesaikan pendidikan: Dr. Joseph Liu dan Dr. Ron Steinfeld.

Jakarta, 2016
Dimaz Ankaa Wijaya

"For my wife Indah and my lovely daughter Sophia

You are my world"

DAFTAR ISI

Daftar Gambar

BAB 1
Pendahuluan

Bitcoin merupakan sistem pembayaran digital yang diperkenalkan oleh Satoshi Nakamoto pada tahun 2008 [1]. Sistem ini merupakan terobosan baru yang memungkinkan orang untuk melakukan transaksi satu sama lain tanpa melalui trusted party (pihak ketiga yang dipercaya seperti bank). Menghapus trusted party di dalam sebuah sistem pembayaran mengharuskan verifikasi atas validitas transaksi keuangan harus dilakukan dengan cara yang berbeda, dan di sinilah peran kriptografi. Karena Bitcoin tidak membutuhkan trusted party, maka sistem ini dapat berjalan dalam sistem peer-to-peer di mana tidak ada satupun yang bertindak sebagai dedicated server, melainkan setiap komputer saling mengirimkan informasi terbaru, sehingga pada akhirnya semua komputer memiliki informasi yang sama. Bitcoin juga berjalan dalam sistem yang terdesentralisasi dengan tidak ada satu pihak pun yang menjadi pusat pengendali sistem.

Untuk memastikan bahwa hanya transaksi sah saja yang diperbolehkan di dalam sistem, Bitcoin mengimplementasikan mekanisme yang disebut dengan Proof-of-Work (PoW), di mana setiap transaksi dihitung nilai hashnya dan dimasukkan ke dalam basis data yang disebut blockchain. Di dalam blockchain terdapat blok-blok yang tersusun atas transaksi-transaksi yang nilai hashnya terhubung satu sama lain membentuk struktur Merkle Tree [2].

Untuk menghubungkan antara satu blok dengan blok yang lain, nilai hash dari blok sebelumnya dimasukkan ke dalam blok berikutnya dan kemudian dihitung nilai hashnya. Nilai hash tersebut harus memenuhi persyaratan tertentu yang disebut difficulty untuk dapat dianggap sebagai blok yang sah. Jika blok tersebut dianggap sah dan memenuhi kriteria, maka jaringan Bitcoin memasukkannya ke dalam blockchain. Usaha pencarian nilai hash yang sesuai persyaratan itulah yang dinamakan Proof-of-Work (PoW), sebab membuktikan bahwa usaha kalkukasi dalam tingkatan tertentu telah dilakukan untuk memenuhi kriteria yang dipersyaratkan.

Bitcoin tidak membutuhkan organisasi pusat apapun dan oleh karena itulah dapat digunakan sebagai sistem pembayaran tanpa bank. Bitcoin tidak membutuhkan campur tangan pemerintah, dan penciptaan koin akan dibatasi hingga sejumlah total 21 juta bitcoin. Koin-koin baru Bitcoin ditambang oleh mereka yang memiliki kekuatan komputasi untuk menghitung nilai hash yang disebut miner (penambang). Untuk setiap blok yang berhasil dibuat, miner akan memperoleh hadiah sebesar 50 bitcoin (50 BTC). Hadiah ini akan dipotong menjadi separuhnya setiap 4 tahun, dan pada pertengahan tahun 2016, hadiah mining akan akan menjadi 12,5 BTC untuk setiap blok yang berhasil ditambang.

Bitcoin tidak hanya populer karena kesederhanaannya, melainkan juga karena karakteristiknya yang menjunjung anonimitas, sehingga pengguna sistem Bitcoin dapat melakukan transaksi tanpa menggunakan identitas. Bagaimanapun juga, anonimitas Bitcoin mulai dipertanyakan. Bahkan anonimitas Bitcoin tidak lagi dianggap murni anonim, melainkan pseudo-anonim, atau anonimitas semu, sebab terdapat beberapa karakteristik yang dapat digunakan untuk mencari informasi tentang para pengguna Bitcoin dan transaksi-transaksi yang dibuatnya.

Karakteristik Bitcoin

Bitcoin mengganti keberadaan trusted party dengan teknik kriptografi untuk memverifikasi transaksi, di mana ide ini pertama kali diusulkan oleh Dai [3] dalam konsep B-money yang serupa e-money. Beberapa teknik kriptografi tersebut di antaranya algoritma hash, public key cryptography, dan digital signature. Beberapa teknik lain yang juga tersedia dalam Bitcoin adalah zero knowledge proof [4] dan Shamir secret sharing [5] untuk pengguna yang menginginkan fitur-fitur tambahan dalam sistem Bitcoin.

Pihak yang Terlibat

Beberapa pihak yang terlibat dalam sistem Bitcoin dapat dijelaskan sebagai berikut.

- **Node**. Node merupakan komputer yang memiliki salinan lengkap atas basis data blockchain. Node Bitcoin terlibat dalam jaringan peer-to-peer Bitcoin dengan merelay (meneruskan) transaksi dari node satu ke node yang lain. Node Bitcoin biasanya dijalankan oleh miner, aktivis Bitcoin, dan vendor-vendor besar yang menerima pembayaran dalam bentuk bitcoin.

- **Miner**. Miner atau penambang adalah mereka yang memiliki sumber daya (waktu, CPU komputer, dan energi listrik) untuk menambahkan transaksi yang dibuat oleh user ke dalam blok dan menambahkan blok tersebut ke dalam blockchain. Miner akan dihadiahi bitcoin baru setiap kali memenangkan sebuah blok valid (sah) yang dapat ditambahkan ke dalam blockchain. Penambang juga akan mendapatkan pembayaran dari ongkos transaksi yang dibayarkan oleh user dari setiap transaksi yang dibuat. Ongkos transaksi ini ditentukan dari ukuran transaksi, di mana semakin besar ukuran transaksi maka ongkos transaksi juga harus semakin besar. Miner memiliki 2 pilihan saat melakukan mining, yaitu dengan melakukan mining sendirian atau bergabung dalam sebuah mining pool.

- **User**. User adalah mereka yang menggunakan Bitcoin sebagai sistem pembayaran dan bertransaksi dengan orang lain. Untuk menjadi seorang user, yang dibutuhkan hanyalah perangkat lunak yang disebut wallet Bitcoin untuk menyimpan informa-

si tentang alamat Bitcoin dan private key pasangannya yang dapat digunakan untuk menggunakan dana yang tersimpan dalam alamat yang dikuasai oleh user tersebut. Wallet Bitcoin juga dapat digunakan untuk mengelola informasi tentang jumlah dana yang dimiliki dari beberapa alamat yang dimiliki, juga untuk mengelola informasi transaksi-transaksi yang pernah dilakukan, demikian pula dengan status transaksi tersebut (confirmed atau unconfirmed).

Pihak-pihak ini merupakan elemen penting di dalam sistem Bitcoin, sehingga sistem ini dapat bekerja dengan baik. Masing-masing komponen memiliki peranannya masing-masing.

Proof of Work

Strategi Proof of Work (PoW) yang digunakan oleh Bitcoin merupakan cara untuk menghindari double spending. Double spending adalah kasus di mana uang yang sama dibelanjakan 2 kali oleh pemilik yang sama, yang tentunya tidak sesuai dengan konsep uang oleh karena itu harus dihindari. PoW merupakan konsep yang pertama kali dipaparkan oleh Adam Back dalam sistem yang disebut dengan Hashcash [6] sebagai sebuah usaha untuk mengurangi email spam (email sampah yang tidak berguna).

Bitcoin juga memiliki sistem timestamp untuk menandai waktu transaksi mana yang lebih dahulu dibandingkan transaksi lain dalam hal transaksi-transaksi tersebut berusaha untuk membelanjakan uang yang sama [1]. Untuk menjaga agar sebuah blok diciptakan setiap 10 menit, maka sistem Bitcoin menentukan sebuah nilai yang disebut difficulty yang dihitung berdasarkan total kekuatan komputasi hash di dalam jaringan Bitcoin [7]. Penyesuaian nilai difficulty dilakukan setiap 2 minggu atau setiap 2016 blok [8] secara otomatis. Nilai difficulty ini menjadi prasyarat seberapa sulit bagi para miner untuk membuat sebuah blok yang valid dan dapat dimasukkan ke dalam blockchain.

Idealnya, 2016 blok diciptakan dalam waktu 2 minggu atau 1.209.600 detik, namun akan ada 2 kondisi di mana nilai difficulty harus disesuaikan. Pertama, jika 2016 blok diciptakan dalam waktu kurang dari 2 minggu, maka nilai difficulty akan meningkat secara proporsional hingga 300%, sehingga 2016 blok berikutnya diharapkan dapat diciptakan dalam waktu tepat 2 minggu dengan asumsi

kekuatan komputasi berada dalam level yang sama. Kedua, jika 2016 blok diciptakan dalam waktu lebih dari 2 minggu, maka nilai difficulty akan menurun secara proporsional hingga 75%.

Di dalam blockchain, hanya block header saja yang dihitung nilai hash nya. Block header ini berukuran 80 byte yang berisi beberapa informasi seperti nilai hash dari blok sebelumnya dan nilai merkle root. Dengan demikian, ukuran transaksi tidak menentukan kecepatan penghitungan nilai hash. Selain block header, di dalam penghitungan nilai hash juga diperlukan sebuah angka yang disebut dengan nonce (number used only once). Alasan penggunaan nonce adalah sebagai berikut. Sebuah nilai block header hanya akan menghasilkan 1 nilai hash saja. Tentunya sangat kecil kemungkinan nilai hash dari block header memenuhi ketentuan difficulty. Oleh karena itu, nonce ditambahkan dalam penghitungan nilai hash untuk mencari nilai mana yang sesuai dengan ketentuan. Nonce akan diubah terus menerus sampai ditemukan nilai hash yang cocok dengan difficulty.

Trustless

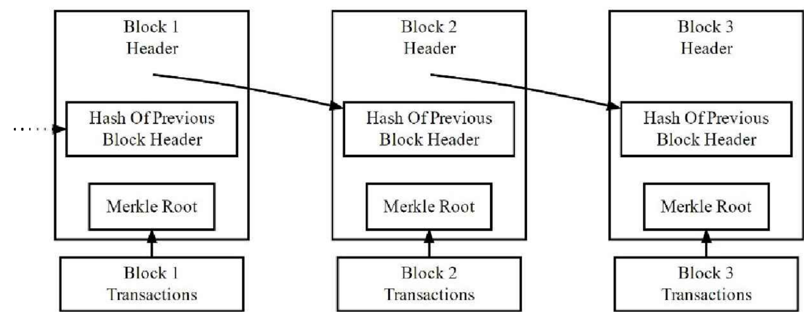
Konsep trustless merupakan kebalikan dari konsep trustful. Konsep trustful merupakan konsep di mana pengguna harus mempercayakan uangnya kepada pihak lain, semisal bank, yang mengontrol sistem finansial. Artinya, bank adalah satu-satunya lembaga yang berwenang mengatur masuk keluarnya uang, sistem keamanan sistem, dan jalannya sistem.

Sementara itu dalam sistem trustless, tidak ada pengendali pusat seperti yang terdapat dalam sistem. Bitcoin menyandarkan sistemnya pada sistem demokrasi mayoritas yang menyerupai sistem voting. Artinya, siapapun yang menjadi mayoritas akan menang. Sistem voting dalam Bitcoin dilakukan oleh para penambang. Para penambang ini akan melakukan voting terhadap rantai blok mana yang akan digunakan, ketika terdapat 2 blok yang dibuat dalam waktu yang bersamaan. Semakin besar kekuatan komputasi yang dimiliki oleh penambang, maka semakin besar juga nilai voting yang dimiliki.

Basis Data Terdistribusi

Salah satu cara yang dapat ditempuh untuk menciptakan sebuah sistem uang digital yang trustless (tanpa pihak pengendali serupa bank sentral) adalah dengan menggunakan basis data terdistribusi,

sehingga semua orang dapat memverifikasi validitas transaksi yang ada dalam sistem [1]. Transaksi-transaksi ini dikelompokkan ke dalam blok yang diciptakan setiap 10 menit oleh para miner. Sebuah blok baru kemudian akan ditambahkan ke posisi paling atas dari rantai blok yang menyusun basis data tersebut. Oleh karena tersusun atas blok-blok yang saling terhubung satu sama lain, basis data terdistribusi ini disebut dengan blockchain.



Gambar 1. Ilustrasi Blockchain. [8]

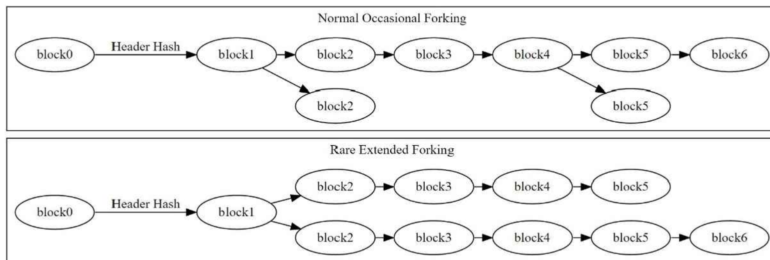
Jika seseorang ingin mengubah sebuah transaksi yang telah dimasukkan dalam sebuah blok yang lalu, maka ia harus mengulangi perhitungan tidak hanya pada blok terkait, tetapi juga blok-blok yang diciptakan sesudah blok tersebut sampai dengan blok terbaru [1]. Oleh sebab itu, semakin dalam blok tersebut, maka usaha untuk mengubah informasi yang ada dalam blok menjadi semakin sulit. Inilah karakteristik yang menarik banyak pihak termasuk perbankan untuk mengembangkan suatu sistem yang memiliki tingkat keamanan yang lebih tinggi dalam mengamankan informasi dari modifikasi yang tidak berizin (unauthorized).

Blockchain sebagai sebuah basis data terdistribusi disalin ke dalam seluruh node yang terhubung dengan jaringan Bitcoin, di mana seluruh node tersebut menyimpan seluruh informasi transaksi Bitcoin dari awal hingga transaksi terbaru, yang membuat sistem Bitcoin menjadi lebih kuat dan terhindar dari permasalahan single point of failure [9], sebab ketika sebuah node tidak beroperasi, node-node lain dapat menggantikan node tersebut dengan informasi basis data yang persis sama.

Selain blockchain, ada sebuah basis data lain yang dikelola oleh node yang disebut Unspent Transaction Output (UTXO) yang menyimpan informasi seluruh uang yang belum dibelanjakan. Informasi UTXO disimpan di dalam memori RAM sehingga dapat dikelola dengan kecepatan tinggi [9].

Forking

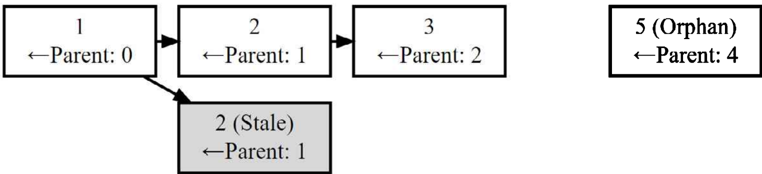
Forking (pencabangan) merupakan kejadian di mana 2 blok dengan nomor blok yang sama tercipta dalam waktu yang bersamaan.



Gambar 2. Forking [8]

Gambar 2 menunjukkan kejadian forking. Berhubung Bitcoin bekerja dalam sistem voting, maka kejadian forking tentu saja bisa terjadi. Saat 2 blok tercipta dalam waktu yang bersamaan, maka masing-masing penambang akan memilih blok mana yang mereka ikuti untuk membuat blok berikutnya.

Dalam kasus normal, biasanya fork hanya terdiri atas 1 blok. Ketika blok tersebut tidak berhasil menjadi bagian dari blockchain, maka blok tersebut akan menjadi stale block yang tidak valid dan dibuang dari blockchain. Sementara blok-blok berikutnya yang tersambung dengan stale block akan menjadi orphan block (blok yatim piatu). Seluruh transaksi yang ada di dalam stale block dan orphan block yang tadinya berstatus confirmed (terkonfirmasi) akan kembali menjadi unconfirmed (belum terkonfirmasi) dan akan dimasukkan ke dalam antrean untuk dimasukkan ke dalam blok berikutnya.



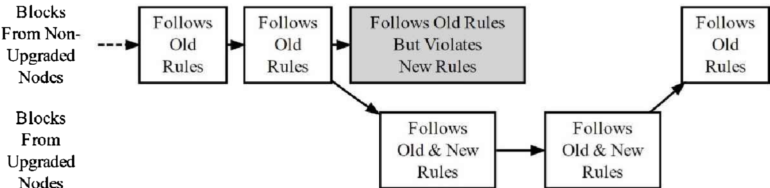
Gambar 3. Stale Block dan Orphan Block [8]

Selain kasus fork di atas, terdapat juga kemungkinan fork karena terjadi perubahan protokol atau sistem Bitcoin yang dihasilkan dari perubahan konsensus. Dalam proses menuju sistem baru, terdapat 2 kemungkinan yang bisa terjadi, yakni soft fork dan hard fork.

Soft Fork

Soft fork merupakan kejadian di mana node-node yang belum melakukan upgrade dapat menerima sebuah blok yang dibentuk dengan menggunakan aturan lama, tetapi node-node yang telah di-upgrade tidak dapat menerima blok baru tersebut karena melanggar aturan baru dalam sistem yang telah diupgrade. Hal ini bisa saja terjadi karena sistem baru memiliki aturan yang lebih ketat dibandingkan dengan sistem lama. Oleh karena itu blok-blok baru yang dibentuk dengan menggunakan sistem baru tetap dapat diterima oleh node yang menjalankan sistem lama.

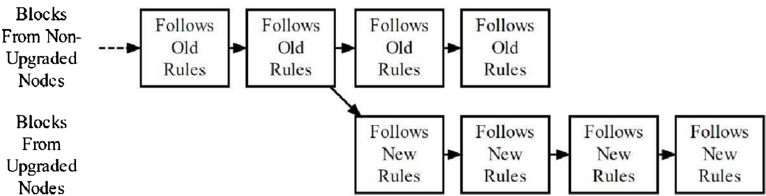
Dengan demikian, meskipun node-node baru tidak mengalami masalah, namun node-node dengan sistem lama akan mengalami problem karena blok yang mereka proses tidak dapat diterima oleh mayoritas node yang memilih untuk mengupgrade sistem mereka ke versi yang lebih baru.



Gambar 4. Soft Fork

Hard Fork

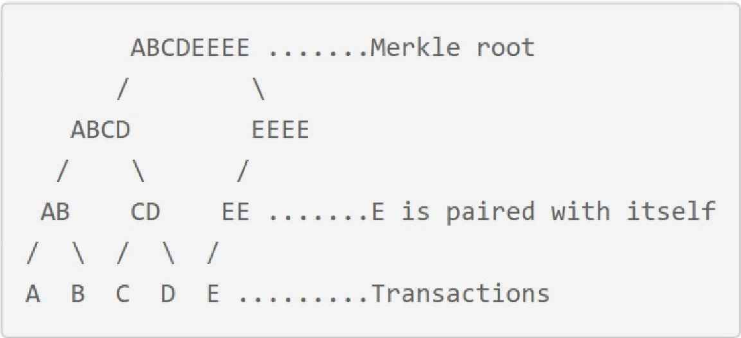
Hal yang berbeda terjadi dalam hard fork. Pada hard fork, node lama dan node baru tidak dapat saling mengupdate informasi blok baru oleh sebab perbedaan protokol yang cukup besar sehingga tidak ada lagi kompatibilitas di antara sistem lama dan sistem baru. Akibatnya dapat muncul cabang yang cukup panjang apabila node lama belum mengupgrade sistem ke versi yang terbaru sesuai dengan konsensus yang telah dicapai.



Gambar 5. Hard Fork

Merkle Tree

Transaksi-transaksi yang terdapat di dalam sebuah blok disusun dalam bentuk merkle tree (pohon merkle) secara bertingkat. Pertama-tama, transaksi-transaksi tersebut akan dpasang-pasangkan kemudian setiap pasang akan dihitung nilai hash nya. Nilai hash tersebut akan dipasangkan kembali dengan nilai hash yang lain untuk mendapatkan sebuah nilai hash yang baru, demikian seterusnya sampai didapatkan sebuah nilai hash yang disebut dengan merkle root.



Gambar 6. Merkle Tree [8]

Berhubung setiap transaksi harus dipasang-pasangkan untuk menghitung merkle root, maka semestinya transaksinya berjumlah genap. Namun jika berjumlah ganjil, transaksi yang tidak memiliki pasangan akan dipasangkan dengan dirinya sendiri.

Terdapat keuntungan penggunaan merkle tree ini dalam hal memverifikasi sebuah blok. Sebagai contoh struktur merkle tree pada Gambar 2, untuk memverifikasi transaksi D, sebuah node tidak perlu menyalin seluruh transaksi A, B, D, dan E melainkan cukup menyalin informasi C, AB, dan EEEE untuk menghasilkan merkle root. Hal inilah yang menyebabkan munculnya node yang tidak memiliki salinan lengkap atas blockchain yang kemudian disebut simplified payment verification (SPV).

Full Node dan Thin Client

Dengan ketiadaan pengendali pusat dalam sistem Bitcoin, maka sistem tersebut dijalankan oleh node dengan mekanisme peer-to-peer. Terdapat 2 jenis node di dalam sistem Bitcoin, yakni full node dan thin client. Full node adalah server yang menyimpan salinan lengkap basis data blockchain yang berisi transaksi Bitcoin dari awal hingga akhir. Sementara thin client memanfaatkan metode SPV dengan hanya memiliki salinan merkle tree tanpa menyimpan transaksi Bitcoin secara keseluruhan.

Kedua jenis node tersebut sama-sama dapat memverifikasi transaksi, hanya saja thin client bergantung pada kejujuran node lain dalam menyediakan informasi merkle tree, sebab ia sendiri tidak dapat memverfikasi kebenaran seluruh transaksi karena tidak menyimpan informasi transaksi-transaksi yang terjadi di dalam sistem Bitcoin. Berbeda dengan full node di mana seluruh transaksi dapat diverifikasi kebenarannya melalui informasi yang tersedia dalam salinan blockchain yang dimiliki. Hanya saja, full node memerlukan sumber daya yang lebih besar dibandingkan dengan thin client berupa kapasitas penyimpanan (harddisk), RAM, dan bandwidth jaringan Internet.

Mainnet dan Testnet

Terdapat 2 jenis jaringan di dalam sistem Bitcoin yang memiliki cara kerja sistem yang sama, namun tidak saling berhubungan. Mainnet merupakan jaringan utama di mana transaksi-transaksi bitcoin dilakukan. Sementara Testnet merupakan jaringan yang dapat di-

gunakan untuk ujicoba. Saat ini Testnet telah memasuki versi ketiga yang disebut dengan Testnet3.

Perbedaan lain yang mencolok di antara 2 jenis jaringan ini adalah awalan alamat Bitcoin yang digunakan. Mainnet menggunakan awalan 1, sementara Testnet3 menggunakan awalan 2 atau 3. Selain itu, bitcoin di dalam Testnet3 (yang disebut test bitcoin atau TBTC) dapat diperoleh secara gratis dari berbagai faucet yang menyediakan, misalnya yang disediakan oleh Coinfaucet.eu di alamat https://testnet.coinfaucet.eu/en/. Meskipun memiliki mekanisme yang sama dengan bitcoin, test bitcoin tidak disarankan untuk diperjualbelikan karena setiap saat dapat dihapus oleh komunitas Bitcoin untuk digantikan dengan versi berikutnya.

User Bitcoin

Bitcoin pertama kali diperkenalkan dalam mailing list kriptografi [10] dan kemudian mendapatkan perhatian yang lebih luas setelah banyak orang menyadari karakteristik yang dimiliki Bitcoin mampu memenuhi kebutuhan akan sistem pembayaran alternatif. Oleh karena popularitas yang terus menanjak, Bohr dan Bashir [11] membuat penelitian untuk mendeskripsikan profil dari para pengguna Bitcoin ini. Menggunakan informasi yang dikumpulkan oleh Smyth [12], mereka melakukan analisis statistik atas informasi demografi seperti umur, lokasi, dan jenis kelamin, termasuk juga ideologi politik yang dianut oleh pengguna.

Hasil dari penelitian ini menyatakan bahwa kebanyakan pengguna Bitcoin pada saat penelitian tersebut dilakukan tinggal di Amerika Serikat dengan usia rata-rata antara 25 sampai 35 tahun. Sementara itu, pandangan politik para user juga menunjukkan bahwa mayoritas pengguna Bitcoin adalah libertarian, yaitu mereka yang menjunjung tinggi kebebasan termasuk di antaranya kebebasan bertransaksi dan cenderung menolak sistem moneter yang dikendalikan oleh pihak perbankan.

Penambangan

Penambangan bitcoin (bitcoin mining) merupakan proses kalkulasi komputer yang dilakukan sebagai syarat validasi data transaksi ke dalam sebuah blok baru. Subbab ini akan membahas sekilas tentang siapa itu penambang, peralatan yang digunakan, dan timbal balik yang diterima oleh para penambang.

Penambang

Para penambang adalah mereka yang menyediakan sumber daya komputasi untuk melakukan kegiatan penambangan. Selain kekuatan komputasi yang besar, para penambang juga harus menyediakan jaringan Internet yang stabil dalam melakukan operasinya. Juga tentunya infrastruktur lain seperti ruang untuk mengelola peralatan, kekuatan listrik yang cukup, dan juga sistem pendingin ruangan untuk mendinginkan mesin-mesin yang ada.

Secara umum 2 jenis penambang. Penambang jenis pertama adalah mereka yang menambang secara sendiri-sendiri atau biasa disebut single mining. Sementara penambang kedua berkelompok sebagai pool mining. Masing-masing tipe penambangan memiliki kelebihan dan kekurangan. Untuk para penambang yang menambang sendiri, seluruh keuntungan akan diperoleh sendiri tanpa harus dibagi kepada orang lain, sementara pool mining membagi-bagi hasil penambangannya kepada seluruh pihak yang berpartisipasi sesuai dengan share pekerjaan yang dikerjakan.

Peralatan

Pada awal diciptakannya Bitcoin, Satoshi Nakamoto diyakini melakukan penambangan dengan menggunakan komputer biasa. Lambat laun, jumlah penambang semakin banyak dan ini menyebabkan kemampuan komputasi semakin besar. Setelah penggunaan CPU tidak lagi menghasilkan banyak bitcoin, maka penambangan berbasis GPU (graphical processing unit) dikembangkan. Lalu, setelah GPU tidak lagi menghasilkan bitcoin yang cukup, mulailah dikembangkan sirkuit khusus berbasis ASIC (Application-Specific Integrated Circuit) untuk melakukan penghitungan dengan lebih efisien.



Gambar 7. Pusat Penambangan Bitcoin di Islandia Milik Genesis Mining

Dengan semakin banyaknya jumlah penambang di dunia, maka dibutuhkan peralatan yang semakin canggih dalam jumlah besar agar memperoleh bitcoin dalam jumlah yang mencukupi untuk membayar ongkos pengadaan alat, sewa ruangan, biaya listrik, dan biaya pendingin udara. Saat ini mayoritas aktivitas penambangan berada di negara China dan negara-negara bersuhu udara rendah. Maraknya penambangan di China disebabkan karena negara ini merupakan pusat produksi sirkuit dan peralatan penambangan, sehingga membangun penambangan di dekat pusat produksi peralatan akan mengurangi biaya transportasi. Selain itu, negara China terkenal dengan infrastruktur yang murah.

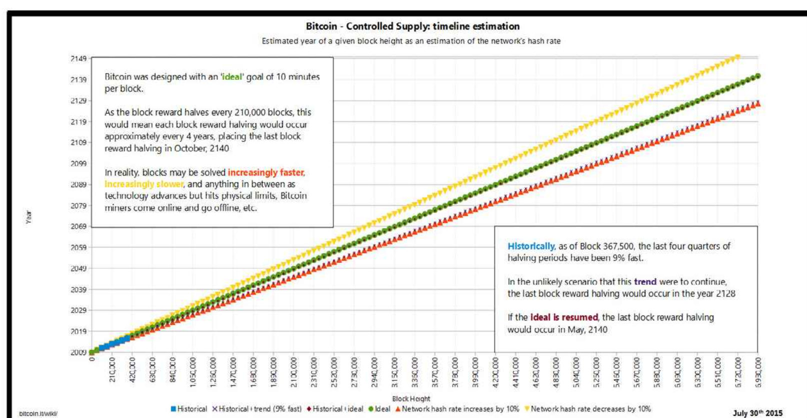
Penempatan peralatan penambangan di negara-negara dingin lebih disebabkan pada potensi penghematan atas biaya pendingin ruangan. Tentu saja peralatan elektronik yang demikian banyak akan menimbulkan energi panas yang besar yang tentunya harus dibuang untuk menjaga agar peralatan elektronik tersebut tetap dapat beroperasi dengan baik.

Mining Reward

Mining reward merupakan hadiah yang diberikan kepada penambang yang berhasil menciptakan sebuah blok baru yang berisi transaksi-transaksi di dalam Bitcoin untuk ditambahkan ke dalam blockchain. Mining reward ini berguna untuk memberi insentif ekonomi

bagi para penambang agar mereka tetap mau menjalankan aktivitas penambangan mereka yang memakan biaya tidak sedikit untuk membayar investasi peralatan, biaya ruangan, energi, dan pemeliharaan. Saat ini mining reward memiliki porsi jauh lebih besar dari jumlah bitcoin yang diterima penambang jika dibandingkan dengan transaction fee (biaya transaksi) yang dibayar oleh pengirim bitcoin saat membuat transaksi Bitcoin.

Mining reward yang akan diberikan kepada penambang memiliki jumlah yang dinamis. Pada awal terbentuknya sistem Bitcoin, mining reward yang ada sebesar 50 BTC. Sementara itu, besaran nilai ini akan berkurang separuhnya setiap 210.000 blok atau kira-kira 4 tahun sekali [13]. Saat ini, mining reward berada dalam level 25 BTC dan akan menjadi 12.5 BTC pada pertengahan tahun 2016. Pemotongan mining reward ini dikenal dengan nama block reward halving. Proyeksi jangka panjang memperkirakan bahwa pada tahun 2140, tidak akan ada lagi mining reward, sehingga para penambang hanya akan mendapat upah dari transaction fee [13].



Gambar 8. Linimasa Estimasi Mining Reward [13]

Mining reward juga merupakan satu-satunya cara untuk membuat bitcoin baru yang ditambahkan ke dalam sirkulasi bitcoin dalam jaringan. Ketika penambang memperoleh mining reward untuk blok yang dihasilkannya, maka total jumlah bitcoin yang beredar juga akan bertambah. Hal ini bermanfaat juga untuk memperluas jangkauan penggunaan bitcoin. Saat mining reward tidak ada lagi, maka total jumlah

bitcoin yang beredar tidak akan bertambah, yang artinya jumlah total bitcoin beredar tidak akan melebihi kisaran angka 21 juta bitcoin [14].

Kelebihan Bitcoin

Sebagai sebuah inovasi teknologi, Bitcoin memiliki beberapa kelebihan dibandingkan dengan sistem finansial konvensional lainnya.

Demokratis

Salah satu karakteristik utama sistem Bitcoin adalah demokratis. Sistem desentralisasi yang ada pada Bitcoin memungkinkan pengambilan keputusan dilakukan dengan lebih demokratis. Hal ini membuat keputusan-keputusan yang dilakukan dalam pengembangan Bitcoin tidak hanya menguntungkan segelintir pengguna saja, melainkan mayoritas pengguna harus mengutarakan persetujuannya. Dengan sistem yang terbuka, siapapun dapat mengkritik, memberi masukan, dan juga berpartisipasi terhadap pengembangan dan pengelolaan sistem Bitcoin.

Kecepatan Transaksi

Transaksi-transaksi Bitcoin diverifikasi dengan menggunakan teknik-teknik kriptografi dan dikonfirmasi ke dalam suatu basis data yang dapat diverifikasi oleh siapapun. Saat ini, sistem Bitcoin dapat menampung hingga 7 transaksi per detik dengan asumsi ukuran blok sebesar 1 MB [15]. Transaksi-transaksi Bitcoin dapat dikonfirmasi dalam waktu sekitar 10 menit. Kecepatan transaksi Bitcoin jauh lebih tinggi jika dibandingkan dengan transaksi menggunakan kartu kredit (Visa, MasterCard, dll) yang membutuhkan waktu beberapa hari untuk difinalisasi.

Rendahnya Biaya Transaksi

Saat ini biaya transaksi Bitcoin berada dalam tingkat yang sangat terjangkau. Nilai biaya transaksi tidak ditentukan dari jumlah bitcoin yang ditransaksikan, melainkan dari ukuran transaksi dalam dihitung dalam satuan byte. Biaya transaksi standar sebesar 10.000 satoshi yang jika dikonversi dalam rupiah sebesar Rp 500 untuk setiap transaksi. Hal ini jauh berbeda jika dibandingkan dengan biaya transaksi kartu kredit yang dapat mencapai 5% dari total nilai uang yang ditransaksikan.

Jangkauan Transaksi

Sistem Bitcoin dapat menjangkau tempat manapun di dunia selama terhubung dengan jaringan Internet. Dengan demikian, transaksi internasional dapat dilakukan dengan mudah dengan menggunakan Bitcoin. Untuk dapat menggunakan sistem Bitcoin, seorang pengguna baru hanya perlu membuat Bitcoin wallet dan mulai mengisi wallet tersebut dengan bitcoin. Tentu saja hal ini sangat berbeda dengan kartu kredit, layanan perbankan, dan bahkan PayPal di mana pengguna harus melakukan registrasi dan verifikasi identitas untuk mulai memanfaatkan sistem keuangan yang disediakan.

Penanganan Atas Double Spending

Double spending merupakan suatu kejadian di mana sebuah bitcoin yang sama digunakan untuk melakukan 2 transaksi yang berbeda. Tentu saja double spending harus dihindari dalam sistem finansial apapun, karena akan merugikan pihak penerima. Sebagai contoh, seorang pengguna A memiliki bitcoin sebesar 0,5 BTC. Ia membeli barang senilai 0,5 BTC dari B, dan membayar barang tersebut dengan uang yang ia miliki. Dalam waktu yang bersamaan, A juga membeli barang senilai 0,3 BTC dari C dengan menggunakan uang yang sama. Tentu saja secara matematis, hal ini tidak dapat dilakukan, karena A setidaknya harus memiliki uang sebesar 0,8 BTC untuk membeli kedua barang dari B dan C.

Untuk menghindari double spending, maka sistem Bitcoin menggunakan basis data terbuka yang dapat diverifikasi oleh siapapun. Dengan basis data terbuka, maka setiap node memiliki basis data yang sama, dan setiap node juga ikut melakukan verifikasi atas setiap transaksi yang dicatat dalam basis data tersebut. Oleh karena itu, meskipun double spending masih dapat terjadi di dalam sistem Bitcoin, namun sistem dapat mendeteksi transaksi-transaksi double spending dengan mudah. Apabila terjadi transaksi double spending, maka sistem Bitcoin hanya akan menerima transaksi pertama yang dikirim ke dalam jaringan, sehingga hanya satu transaksi saja yang akan dikonfirmasi oleh sistem Bitcoin.

Kekurangan Bitcoin

Selain memiliki kelebihan, tentunya Bitcoin memiliki kekurangan. Berikut ini akan dibahas beberapa permasalahan yang terdapat pada sistem Bitcoin.

51% Attack

Serangan yang dinamai 51% attack merupakan tipe serangan dengan cara menguasai mayoritas (lebih dari 50%) kekuatan komputasi dalam jaringan Bitcoin sehingga dapat melakukan apa saja terhadap blockchain. Meskipun tipe serangan ini tidak mustahil dilakukan, namun tentu saja membutuhkan biaya yang sangat besar untuk melakukan serangan ini terhadap sistem Bitcoin yang ada sekarang. Hal ini disebabkan karena jumlah kekuatan komputasi di dalam sistem Bitcoin sudah sangat besar, sehingga diperlukan investasi miliaran dolar AS untuk menjadi mayoritas di dalam sistem Bitcoin.

Denial of Service

Denial of Service (DoS) sebenarnya merupakan tipe serangan yang umum dilakukan terhadap server-server di dalam jaringan Internet. Tipe serangan ini dapat juga dilakukan kepada node Bitcoin yang menyediakan kopian blockchain. DoS dilakukan dengan cara membanjiri server dengan ribuan bahkan jutaan permintaan data, sehingga menyebabkan server tersebut kehabisan sumber daya untuk menangani permintaan data yang masuk.

Meskipun pada awal-awal pengembangan sistem Bitcoin, serangan DoS banyak dialami oleh node server, saat ini aplikasi Bitcoin telah memiliki beberapa metode untuk mengurangi kemungkinan serangan DoS [16].

Sybil Attack

Sybil Attack merupakan tipe serangan yang lebih efisien dibandingkan 51% attack. Sybil attack berusaha membanjiri jaringan Bitcoin dengan node-node yang dikuasainya. Klien-klien yang terhubung dengan node jahat tersebut dapat dipastikan mendapatkan informasi blockchain yang salah yang disediakan oleh node jahat tersebut. Dengan demikian, potensi adanya double spending bisa terjadi.

Namun demikian, sistem Bitcoin telah mengantisipasi hal ini dengan menyediakan daftar node terpercaya, dan juga memberikan hukuman atas node yang terbukti bertindak jahat [16].

Selfish Mining

Selfish mining adalah penambangan yang dilakukan oleh seorang atau sekelompok penambang yang memiliki kekuatan komputasi

yang cukup besar yang tidak segera mempublikasikan blok baru yang berhasil diciptakannya. Blok baru tersebut akan disimpan cukup lama agar penambang tersebut punya waktu lebih banyak untuk menciptakan blok-blok baru. Setelah jumlah blok yang diciptakan dirasa cukup, maka penambang tersebut akan mempublikasikan blok-blok tersebut sekaligus ke dalam jaringan Bitcoin.

Selfish mining dapat menimbulkan kerugian bagi penambang lain yang berhasil menciptakan blok baru, namun pada akhirnya blok baru tersebut menjadi stale blok atau orphan block, sehingga tidak mendapatkan bitcoin baru. Selfish mining tentu saja menguntungkan para penambang yang melakukannya, namun demikian cukup berisiko untuk melakukan selfish mining, karena tidak dapat memastikan keberhasilan metode ini.

Transaction Malleability

Transaction malleability merupakan sebuah kejadian di mana sebuah transaksi sedikit diubah datanya tanpa mengubah makna transaksi tersebut. Transaction malleability menyebabkan perubahan Transaction ID (TxID) atas transaksi tersebut. Transaction malleability sebenarnya bukan merupakan kelemahan dari sistem Bitcoin karena merupakan karakteristik dari teknik-teknik yang digunakan dalam menyusun sistem. Namun hal ini menjadi problem ketika TxID menjadi patokan dalam menentukan apakah dana yang dikirimkan oleh pihak pembayar kepada pihak penerima berhasil dikonfirmasi oleh sistem Bitcoin.

Terdapat beberapa jenis transaction malleability yang terdapat di dalam sistem Bitcoin, yakni signature malleability dan scriptSig malleability [17]. Signature malleability terjadi ketika terjadi perubahan atas digital signature, meskipun digital signature yang baru sama validnya dengan digital signature yang lama. Hal ini terjadi karena karakteristik ECDSA di mana (r, s) sama dengan (r, -s(mod N)). Sementara scriptSig malleability memanfaatkan penyusunan script yang cukup longgar di dalam sistem Bitcoin, di mana dapat disusun 2 script yang berbeda namun memiliki makna yang sama.

Transaction malleability sebenarnya bukan termasuk salah satu kelemahan dalam sistem Bitcoin, namun menjadi terkenal lantaran Mark Karpeles, CEO Mt. Gox, menyalahkan transaction malleability atas kebangkrutan Mt. Gox dan hilangnya ratusan ribu bitcoin milik

para klien Mt. Gox. Meski demikian, para pakar meragukan kebenaran klaim tersebut [18].

Konsumsi Energi

Isu pemborosan energi atas kegiatan penambangan dikritik oleh Ben Laurie, pendiri Apache Software Foundation dalam tulisannya yang mengemukakan inefisiensi sistem Bitcoin [19]. Ketimbang melakukan aktivitas penambangan untuk bersaing dalam menciptakan blok baru, Laurie mengusulkan agar bitcoin baru dapat dibagi rata kepada para anggota sistem demi efisiensi penggunaan energi.

Kegiatan penambangan bitcoin merupakan aktivitas yang membutuhkan banyak energi listrik untuk menyalakan ribuan bahkan jutaan peralatan elektronik untuk melakukan kalkulasi matematis. Energi listrik juga digunakan untuk menghidupkan mesin pendingin ruangan untuk menjaga agar peralatan elektronik tersebut dapat digunakan dalam waktu yang lama. Meskipun hal ini tampak buruk dari segi ekologi, namun konsumsi energi yang besar dapat pula melindungi sistem Bitcoin dari serangan 51% attack, sebab membuktikan bahwa untuk melakukan 51% attack diperlukan investasi yang sangat besar. •

BAB 2 LATAR BELAKANG TEKNOLOGI

Sebagaimana telah dijelaskan sebelumnya bahwa sistem Bitcoin tidak mengenal trusted party yang mengatur transaksi menggunakan bitcoin, Bitcoin sangat mengandalkan teknik kriptografi yang akan dibahas pada bagian ini. Selain itu akan dibahas juga hal-hal detail mengenai Bitcoin, yakni komponen-komponen yang menyusun transaksi Bitcoin, jenis-jenis transaksi, dan bagaimana sebuah transaksi dibuat.

Teknologi Kriptografi

Teknologi kriptografi merupakan salah satu tulang punggung sistem Bitcoin, sebab ketiadaan trusted party menyebabkan Bitcoin sangat bergantung pada kalkulasi dan teknik-teknik kriptografi. Berikut ini akan dibahas beberapa teknologi yang dipakai pada sistem Bitcoin, di antaranya public key cryptography, digital signature, dan algoritma hash.

Public Key Cryptography

Public key cryptography merupakan sistem yang dikembangkan untuk menjawab permasalahan atas pertukaran kunci rahasia dari private key cryptography [20]. Pada private key cryptography, sebagai contoh Alice dan Bob ingin berkomunikasi dengan aman, mereka harus bertukar informasi berupa kunci rahasia K. Ketika Alice ingin mengirim pesan M pada Bob, ia mengenkripsi pesan tersebut dengan menggunakan rumus

$$Q = E(M,K)$$

dengan Q adalah pesan terenkripsi, E adalah algoritma enkripsi, M adalah pesan yang akan dikirimkan, dan K adalah kunci rahasia. Pesan terenkripsi Q akan dikirim oleh Alice kepada Bob, dan untuk mendapatkan pesan M, maka Bob harus mendekripsi pesan Q dengan menggunakan rumus

$$M = D(Q,K)$$

dengan M adalah pesan, D adalah algoritma dekripsi, Q adalah pesan terenkripsi, dan K adalah kunci rahasia yang hanya diketahui oleh Alice dan Bob. Permasalahan dari sistem ini adalah kesulitan mendistribusikan kunci rahasia, sebab sebelum memulai komunikasi dengan menggunakan private key cryptography, Alice dan Bob sudah harus mendapatkan kunci rahasia yang sama.

Untuk menangani permasalahan ini, Diffie and Hellman [21] mengusulkan ide digunakannya public key cryptography yang menggunakan 2 kunci alih-alih menggunakan 1 kunci sebagaimana digunakan dalam private key cryptography. Public key cryptography memperkenalkan kunci publik dan kunci privat. Pada public key cryptography,

apabila diketahui kunci publik, maka kunci privat tidak akan mudah dihitung berdasarkan kunci publik yang telah diketahui tadi. Oleh karena itu, kunci publik dapat dengan bebas diinformasikan kepada orang lain, sementara kunci privat harus disimpan dengan baik oleh pemegangnya. Dengan menggunakan skema ini, maka jika Alice ingin mengirim pesan M kepada Bob, maka Alice menggunakan kunci publik milik Bob untuk mengenkripsi pesan dan mengirimkannya pada Bob. Kemudian setelah pesan terenkripsi tersebut diterima oleh Bob, maka hanya Bob yang dapat mendekripsi pesan tersebut dengan menggunakan kunci privat miliknya.

Salah satu metode kriptografi yang bisa digunakan di antaranya RSA yang didasarkan pada faktorisasi integer [22]. RSA diciptakan oleh 3 orang pakar kriptografi yakni Ron Rivest, Adi Shamir, dan Leonard Adleman. RSA sendiri dinamai berdasarkan inisial nama ketiga penemu tersebut, yaitu Rivest, Shamir, dan Adleman. RSA merupakan implementasi pertama dari skema public key cryptography, yang kemudian berkembang pula skema-skema lain seperti ElGamal [23] dan Elliptic Curve Cryptography.

Elliptic Curve Cryptography

Bitcoin mengimplementasikan Elliptic Curve Cryptography (ECC) karena beberapa alasan. Jika dibandingkan dengan skema public key cryptography lain, ECC memiliki ukuran kunci yang kecil, sebagaimana tampak pada gambar. Karena ukuran yang kecil inilah, komputasi dapat dilakukan lebih cepat dibandingkan jika menggunakan ukuran kunci yang lebih besar [9]. ECC dengan ukuran kunci 256 bit yang digunakan dalam sistem Bitcoin memiliki tingkat keamanan yang setara dengan kunci 128 bit pada symmetric cryptography (private key cryptography) dan 3072 bit pada sistem RSA.

Pada ECC, kunci publik dapat dianggap sebagai sebuah titik, dan kunci privat dianggap sebagai langkah-langkah dari sebuah titik awal yang disebut generator ke titik yang ditunjuk oleh kunci publik. Artinya, akan mudah menghitung kunci publik apabila diketahui kunci privatnya, namun akan sulit menghitung kunci privat jika diketahui kunci publiknya. Untuk implementasi sistem Bitcoin, Satoshi memilih untuk mengimplementasikan Secp256k1 [24] dengan parameter sebagai berikut.

Digital Signature

Digital signature atau tanda tangan digital merupakan sebuah cara untuk membuktikan identitas seorang user yang dapat diverifikasi oleh orang lain. Digital signature memanfaatkan public key cryptography. Digital signature dapat dimanfaatkan dalam contoh berikut. Misalnya Alice dan Bob berkomunikasi menggunakan media komunikasi yang tidak aman (orang lain bisa ikut membaca pesan). Pertama, Alice membuat sepasang kunci yang terdiri atas kunci privat dan kunci publik. Kemudian, Alice mengirimkan kunci publik tersebut kepada Bob dan menyimpan kunci privat. Kemudian, Alice membuat digital signature dengan menggunakan algoritma digital signature, kunci privat, dan pesan yang akan disampaikan. Digital signature ini akan digabungkan dengan pesan yang dikirimkan pada Bob. Bob kemudian akan memverifikasi digital signature tersebut menggunakan kunci publik milik Alice. Apabila hasil verifikasi tersebut sama dengan pesan yang dikirimkan, maka terbukti bahwa Alice lah yang mengirim pesan, sebab hanya Alice yang memiliki kunci privat untuk membuat digital signature yang valid.

Bitcoin menggunakan skema ECDSA (Elliptic Curve Digital Signature Algorithm) untuk keperluan kalkukasi digital signature.

Fungsi Hash

Fungsi Hash merupakan algoritma yang akan menghitung sebuah data dengan ukuran berapapun ke dalam sebuah nilai dengan ukuran tertentu. Terdapat beberapa karakteristik yang dimiliki oleh fungsi hash, yakni hasil kalkulasi memiliki ukuran (dalam bit) tertentu, dan juga kalkukasi membutuhkan waktu yang singkat untuk menghasilkan output. Juga, sebuah fungsi hash harus menghasilkan nilai yang sama apabila nilai inputnya sama. Pada sistem Bitcoin, terdapat beberapa fungsi hash yang digunakan dalam protokolnya.

SHA256

SHA256 merupakan fungsi hash yang distandardisasi oleh NIST (National Institute of Standards and Technology) pada tahun 2001 sebagai salah satu dari keluarga SHA generasi kedua. SHA256 akan menghasilkan output dengan ukuran 256 bit. Bitcoin menggunakan algoritma SHA256^2 atau SHA256 ganda, artinya algoritma SHA256 dilakukan 2 kali. Hal ini dilakukan karena terdapat sebuah permas-

alahan pada algoritma ini yang disebut length extension attack. SHA256^2 digunakan salah satunya pada proses mining blok Bitcoin.

RIPEMD160

RIPEMD (RACE Integrity Primitives Evaluation Message Digest) merupakan algoritma hash yang dikembangkan berdasarkan MD4. Algoritma ini menghasilkan nilai hash dengan ukuran 160 bit. RIPEMD160 digunakan untuk membuat alamat Bitcoin yang dikalkulasi berdasarkan kunci publik [25].

Shamir Secret Sharing

Shamir [5] memperkenalkan sebuah mekanisme secret sharing atau berbagi rahasia dengan menggunakan persamaan matematika yangn disebut skema treshold (k,n) di mana k < n. Pada skema ini, sebuah nilai rahasia D akan dipecah ke dalam n bagian dan dapat dihitung kembali nilainya hanya dengan menggunakan bagian sejumlah k dari total n yang tersedia. Untuk menggunakan skema ini, perlu digunakan persamaan polinomial derajat k-1 sebagai berikut.

$$q(x) = a_0 + a_1x + \cdots + a_{k-1}x^{k-1}$$

Di mana adalah nilai rahasia yang akan dihapus ketika nilai-nilai sebanyak n telah selesai dihitung. Pada skema di atas, q(1), q(2), ..., q(n) dihitung dan didistribusikan ke pihak-pihak yang berbeda. Apabila ada pihak sejumlah k setuju untuk menghitung kembali nilai rahasia D, maka nilai D dapat dihitung dengan menggunakan interpolasi polinomial. Dengan menggunakan q(x) sejumlah k, maka dapat dihitung untuk kemudian dapat dihitung.

Shamir Secret Sharing dapat digunakan misalnya dalam hal pengamanan private key wallet online, private key tersebut dipecah ke dalam beberapa bagian, dan untuk mendapatkannya kembali harus menyatukan bagian-bagian tersebut. Cara ini meningkatkan keamanan sehingga pihak lain yang hanya menguasai 1 bagian tidak dapat mengambil private key dan mengambil alih dana yang ada pada alamat Bitcoin yang terasosiasi dengan private key yang ada.

Secure Multiparty Computation

Ide atas secure multiparty computation (MPC) pertama kali dikemukakan oleh Andrew C. Yao untuk membuat kalkulasi bersama-sama beberapa partisipan dengan menyembunyikan input dari para partisipan tersebut [26]. MPC dapat digunakan untuk memecahkan beberapa problem, sebagai contoh persoalan tentang 2 orang miliarder yang ingin mengetahui siapa yang lebih kaya di antara keduanya tanpa harus memberi informasi jumlah kekayaan keduanya.

MPC dapat juga digunakan di dalam sistem Bitcoin, di antaranya skema pertaruhan tanpa menggunakan trusted party. Dengan menggunakan MPC, pemenang dapat ditentukan tanpa peluang para peserta berbuat curang [27].

Zero Knowledge Proof

Zero Knowledge Proof (ZKP) merupakan sebuah metode untuk membuktikan kepemilikan sebuah informasi tanpa menunjukkan informasi tersebut [9]. Kedua pihak yang terlibat dalam skema ZKP disebut prover dan verifier. Prover memiliki jawaban atas sebuah permasalahan, dan verifier bertugas memberi pertanyaan terkait dengan permasalahan tersebut kepada prover dan kemudian memverifikasi jawaban yang diberikan oleh prover.

Ide ZKP ini dapat diterapkan di dalam sistem Bitcoin dengan menggunakan fungsi hash untuk mengikat transaksi dengan menggunakan password tanpa harus menginformasikan password tersebut kepada pihak lain [4]. ZKP yang diterapkan membuat verifier tidak memiliki informasi terhadap password, namun prover juga tidak dapat mengubah nilai hash karena telah diinformasikan kepada verifier.

Base58Check Encoding

Base58Check merupakan sebuah metode untuk merepresentasikan deretan nilai byte ke dalam bentuk yang mudah dibaca manusia [28]. Base58 merupakan modifikasi dari metode lain dengan menghilangkan beberapa karakter yang mungkin terlihat sama pada beberapa font, seperti 0 (nol) dan O (huruf o besar), I (huruf I besar) dan l (huruf L kecil). Byte tersebut akan diperlakukan seperti big integer dan kemudian dibagi 58. Sisa pembagian (modulus) itu akan dikonversi ke dalam sebuah karakter. Berikut diagram konversi Base58.

Value	Character	Value	Character	Value	Character	Value	Character
0	1	1	2	2	3	3	4
4	5	5	6	6	7	7	8
8	9	9	A	10	B	11	C
12	D	13	E	14	F	15	G
16	H	17	J	18	K	19	L
20	M	21	N	22	P	23	Q
24	R	25	S	26	T	27	U
28	V	29	W	30	X	31	Y
32	Z	33	a	34	b	35	c
36	d	37	e	38	f	39	g
40	h	41	i	42	j	43	k
44	m	45	n	46	o	47	p
48	q	49	r	50	s	51	t
52	u	53	v	54	w	55	x
56	y	57	z				

Gambar 9. Base58Check [28]

Transaksi Bitcoin

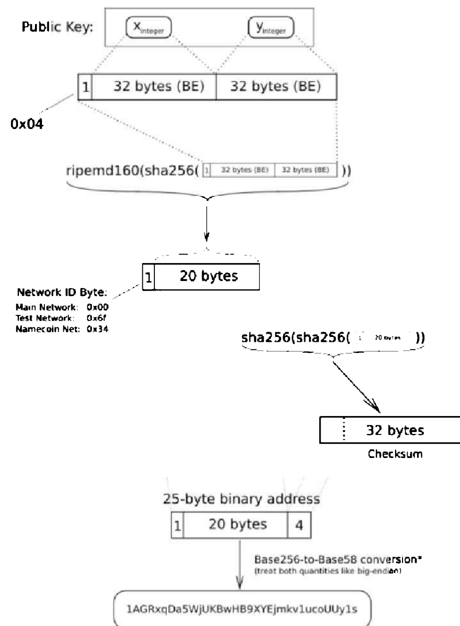
Transaksi Bitcoin merupakan gabungan dari informasi yang terkait satu sama lain, di mana sebuah transaksi setidaknya memiliki 1 input dan 1 output. Sebuah input dari sebuah transaksi merupakan output dari transaksi sebelumnya. Transaksi Bitcoin sebenarnya merupakan teka-teki matematika, di mana bagian input merupakan jawaban dari pertanyaan yang disampaikan di bagian output dari transaksi lain yang diacunya, sementara bagian output dari transaksi tersebut merupakan pertanyaan yang harus dijawab pada transaksi berikutnya [9]. Sebuah transaksi Bitcoin terdiri atas input transaksi (transaction input) yang disebut TxIn dan output transaksi (transaction output) yang disebut TxOut. Dengan demikian, seluruh transaksi dalam Bitcoin merupakan informasi yang saling terkait, yang dapat dijelaskan melalui diagram berikut.

Gambar 10. Transaksi Bitcoin [8]

Selain terdiri atas setidaknya 1 input dan 1 output, transaksi Bitcoin juga menentukan jumlah bitcoin yang ditransaksikan dari input ke masing-masing output. Selisih antara nilai bitcoin yang ada pada sisi input dan nilai bitcoin yang ada pada sisi output merupakan ongkos transaksi yang dibayarkan kepada miner yang akan memasukkan transaksi ini ke dalam blockchain.

Alamat Bitcoin

Alamat Bitcoin dihasilkan dari kunci publik Elliptic Curve [29]. Proses kalkulasi dari kunci publik menjadi alamat Bitcoin dapat dijelaskan melalui diagram berikut.



Gambar 11. Proses konversi Kunci Publik Menjadi Alamat Bitcoin [29]

Langkah-langkahnya dapat dijelaskan sebagai berikut.

- Buat sepasang kunci privat dan publik ECC.
- Menggunakan kunci publik, konstruksikan 65 byte data yang terdiri atas:
  - Bagian pertama terdiri atas 1 byte yaitu 0x04. Informasi ini adalah penanda untuk titik Elliptic Curve (EC) yang tidak terkompresi. Untuk menggunakan titik terkompresi gunakan 0x02 atau 0x03. Mode terkompresi maksudnya hanya menggunakan nilai X tanpa nilai Y, sementara nilai Y nanti akan dihitung dari nilai X.
  - Bagian kedua terdiri atas 32 byte koordinat X dari titik EC.
  - Bagian ketiga terdiri atas 32 byte koordinat Y dari titik

EC.

- Hitung nilai hash menggunakan operasi SHA256 dari data yang diperoleh dari langkah b.
- Hitung nilai hash menggunakan operasi RIPEMD dari nilai hash yang diperoleh dari langkah c. Hasilnya merupakan 20 byte data.
- Gabungkan byte versi dengan langkah (d). Byte versi merupakan penanda alamat, di mana terdapat beberapa versi yang tersedia, yakni 0x00 untuk Mainnet dan 0x6f untuk Testnet.
- Hitung nilai hash menggunakan SHA256^2 dari data yang diperoleh pada langkah (e). Empat byte pertama akan digunakan sebagai checksum.
- Gabungkan data dari langkah (e) dengan 4 byte pertama dari langkah (f).
- Hitung nilai Base58Check data dari langkah (g) dan inilah alamat Bitcoin yang dihasilkan.

Alamat Bitcoin yang digunakan pada transaksi Pay-to-Address ini serupa dengan nomor rekening bank. Pengirim akan membuat output transaksi menjadi hanya bisa dipakai oleh siapapun yang memiliki kunci privat dari alamat Bitcoin tujuan, yang dalam hal ini tentunya penerima. Oleh karena itu, meskipun alamat Bitcoin boleh diinformasikan ke setiap orang, hanya sang pemilik alamat Bitcoin yang memegang kunci privat sajalah yang dapat menggunakan uang yang ada dalam alamat tersebut. Dengan menggunakan teknik kriptografi, ketika sang pemilik alamat Bitcoin ingin menggunakan uangnya, sistem Bitcoin akan memeriksa apakah si pemilik memiliki kunci privat yang cocok dengan alamat Bitcoin, yang sebenarnya merupakan kunci publik dari skema ECC.

Script

Seperti yang telah dibahas sebelumnya, transaksi Bitcoin terdiri atas input dan output. Input dan output ini sebenarnya disusun dari skrip Bitcoin. Skrip ini mirip dengan bahasa pemrograman yang disebut dengan Forth [30].

Pada transaksi Bitcoin umumnya, <scriptSig> digunakan sebagai TxIn, sementara <scriptPubKey> berfungsi sebagai TxOut. Script ini dievaluasi dari atas ke bawah sebagai stack dengan metode LIFO (Last

In First Out). Script pada Bitcoin disebut sebagai tidak Turing-complete karena tidak mengizinkan iterasi (perulangan) untuk menghindarkan sistem dari serangan yang menggunakan kalkulasi kmopleks, oleh karena itu tipe-tipe operasi sangat dibatasi yang didefinisikan dalam OpCode yang direpresentasikan dalam bilangan hexadecimal [30].

Pay To Address

Pay To Address (P2A) atau disebut juga Pay To Public Key Hash (P2PKH) merupakan metode transaksi Bitcoin yang paling umum dilakukan, yakni metode pembayaran yang ditujukan kepada alamat Bitcoin [31] yang merupakan nilai hash dari public key. Untuk dapat menggunakan metode P2PKH, si calon penerima uang harus memberitahukan alamat Bitcoin kepada pembayar untuk kemudian si pembayar membuat sebuah transaksi yang mentransfer uang dari alamat Bitcoin miliknya kepada alamat Bitcoin si penerima.

Informasi <ScriptPubKey> pada metode P2PKH disusun seperti struktur berikut.

OP_DUP OP_HASH160 <hashPubKeyHex> OP_EQUALVERIFY OP_CHECKSIG

Skrip di atas ditempatkan pada TxOut, yang akan memverifikasi TxIn yang akan menggunakan dana pada transaksi tersebut. TxIn yang dibutuhkan untuk menyelesaikan persoalan di atas tersusun atas <sig> dan <pubKey>, di mana <sig> merupakan digital signature yang dibuat oleh pemilik kunci privat atas transaksi yang menggunakan dana. Sementara <pubKey> merupakan kunci publik yang berasosiasi dengan alamat Bitcoin yang bersangkutan. Berikut adalah metode evaluasi script pada Bitcoin.

Stack	Script	Description
Empty.	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	scriptSig and scriptPubKey are combined.
<sig> <pubKey>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Constants are added to the stack.
<sig> <pubKey> <pubKey>	OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is duplicated.
<sig> <pubKey> <pubHashA>	<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top stack item is hashed.
<sig> <pubKey> <pubHashA> <pubKeyHash>	OP_EQUALVERIFY OP_CHECKSIG	Constant added.
<sig> <pubKey>	OP_CHECKSIG	Equality is checked between the top two stack items.
true	Empty.	Signature is checked for top two stack items.

Gambar 12. Evaluasi Script Pay-to-Address [30]

Langkah-langkahnya adalah sebagai berikut.

- Informasi <sig> dan <pubKey> dimasukkan ke dalam stack dengan metode LIFO.
- Operasi pertama adalah OP_DUP, yaitu melakukan duplikasi atas data terakhir dalam stack, yaitu <pubKey>. Dengan demikian, dalam stack tersedia data <sig> <pubKey> <pubKey>.
- Operasi berikutnya adalah OP_HASH160, yaitu menghitung nilai hash dengan metode RIPEMD160 atas data terakhir dalam stack. Setelah operasi ini dilakukan, dalam stack tersedia data <sig> <pubKey> <pubKeyHash>.
- Data <hashPubKeyHex> dimasukkan ke dalam stack, sehingga data dalam stack berubah menjadi <sig> <pubKey> <pubKeyHash> <hashPubKeyHex>.
- Operasi OP_EQUALVERIFY akan mengecek apakah 2 data terakhir pada stack memiliki nilai yang sama. Jika kedua data tersebut sama, maka dihilangkan dari stack.
- Operasi OP_CHECKSIG akan mengecek validitas digital signature <sig> dengan menggunakan <pubKey> yang tersedia.

Apabila operasi terakhir valid, maka script ini akan menghasilkan nilai akhir True (benar), namun jika tidak valid maka hasil akhirnya adalah False (salah).

Pay To Public Key

Transaksi Pay To Public Key (P2PK) bekerja dengan cara yang mirip dengan Pay To Address. Perbedaannya, pada P2PK hanya terdiri atas langkah terakhir (f) dari prosedur yang ada pada P2A. Pada P2PK, <scriptPubKey> disusun seperti struktur berikut.

<pubKey> OP_CHECKSIG

Dengan demikian, untuk menggunakan dana yang tersedia, hanya perlu meletakkan informasi <sig> yang merupakan digital signature, pada bagian <scriptSig>. Evaluasi script ini dapat dijelaskan sebagai berikut.

- Data <sig> dimasukkan ke dalam stack.
- Data <pubKey> dimasukkan ke dalam stack.
- Operasi OP_CHECKSIG akan mengecek validitas digital signature <sig> dengan menggunakan <pubKey> yang tersedia.

Meskipun operasi pada transaksi Pay To Public Key tampak lebih sederhana jika dibandingkan dengan skema Pay to Address, tetapi skema ini akan menghasilkan data transaksi yang lebih besar ketimbang skema Pay to Address, dengan demikian biaya transaksi akan membengkak pula. Tambahan lagi, penelitian menyatakan bahwa skema ini lebih rentan serangan yang dilakukan menggunakan komputer kuantum [9]. Skema ini juga sudah tidak dapat dipakai lagi kecuali jika digabungkan dalam skema Pay To Script Hash (P2SH).

Pay To Script Hash

Pay To Script Hash (P2SH) merupakan metode lain untuk bertransaksi dalam sistem Bitcoin [32]. Metode ini didefinisikan sebagai metode standard dengan spesifikasi BIP16 yang menjelaskan detail dari P2SH [33]. Dengan metode ini, user Bitcoin dapat mengkonstruksikan sebuah script sebagai persyaratan sebelum dapat menggunakan uang yang dibayarkan. Tujuan utama dukungan terhadap P2SH dalam sistem Bitcoin adalah untuk metode multisignature tanpa perlu mendeskripsikan seluruh detailnya dalam ScriptPubKey. Dengan menggunakan P2SH, pengirim uang hanya perlu menuliskan nilai hash dari script yang diinginkan pada ScriptPubKey dan oleh karena itu akan membuat biaya transaksi menjadi lebih murah bagi pengirim. Transaksi P2SH ditandai dengan format berikut.

OP_HASH160 <20 byte hash> OP_EQUAL

Sementara script yang memiliki nilai hash yang telah ditentukan

tersebut harus dapat dipenuhi oleh penerima. Selain script, penerima juga harus menyajikan input-input yang diminta pada P2SH. Input-input tersebut kemudian dievaluasi untuk menentukan apakah memenuhi kriteria script atau tidak.

Null Script

Null Script merupakan salah satu fitur dalam transaksi Bitcoin yang dananya tidak dapat digunakan. Null Script merupakan salah satu cara untuk mengirim pesan menggunakan sistem Bitcoin. Null Script dapat disusun menggunakan struktur sebagai berikut.

OP_RETURN <data>

Dengan data bisa berupa apa saja dengan format BASE16.

Multisignature

Multisignature merupakan salah satu fitur Bitcoin di mana user dapat membuat sebuah transaksi yang membutuhkan lebih dari 1 digital signature untuk dapat digunakan [34]. Fitur ini dapat digunakan dalam skema escrow. Escrow merupakan konsep di mana pihak ketiga menjadi penengah dalam sebuah transaksi ketika terjadi permasalahan baik dari sisi pembeli maupun penjual. Escrow dapat digunakan untuk melindungi pembeli maupun penjual. Dengan menggunakan transaksi 2-of-3 multisignature, apabila terjadi dispute (permasalahan) antara pembeli dan penjual, maka pihak escrow dapat ikut campur dengan menentukan siapa yang mendapatkan uangnya. Apabila penjual telah mengirimkan produk yang dipesan namun pembeli tidak ingin membayar, maka pihak escrow dapat menandatangani transaksi sehingga penjual tetap mendapatkan uang pembayarannya.

Apabila pembeli tidak menerima produk yang dipesan dan penjual tidak ingin mengembalikan uang yang telah dibayarkan, maka pihak escrow dapat membantu pembeli untuk mendapatkan kembali uangnya. Multisignature dapat pula digunakan dalam skema perusahaan untuk dapat mengelola dana dengan lebih bertanggungjawab di mana untuk membelanjakan dana tersebut diperlukan lebih dari 1 pihak untuk memberikan tandatangannya. Skema multisignature dapat dideskripsikan dengan struktur sebagai berikut.

m <pubkey 1> <pubkey 2> ... <pubkey n> n OP_CHECKMULTISIG

dengan m ≤ n ≤ 3.

Multisignature dengan struktur seperti di atas dapat diselesaikan dengan struktur sebagai berikut.

0 <signature 1> <signature 2> ... <signature n>

Nilai 0 merupakan data dummy yang diperlukan sebab terdapat sebuah bug pada implementasi operasi OP_CHECKMULTISIG di mana bug tersebut akan membuang 1 data lebih banyak daripada yang seharusnya. Implementasi sistem Bitcoin yang sekarang tidak mengizinkan data dummy selain nilai 0 pada struktur di atas.

Hash-locked Transaction

Hash-locked transaction (HLT) merupakan skema transaksi Bitcoin yang pertama kali diusulkan oleh Tiernan [35]. Skema ini merupakan gabungan antara skema lain seperti P2PKH atau P2PK dengan penggunaan password dalam transaksi. Dengan demikian, selain harus menyajikan digital signature yang benar, penerima juga harus menyebutkan password yang benar.

Dalam skema ini, kriteria password yang benar adalah password yang memiliki nilai hash yang telah ditentukan sebelumnya oleh pembayar. Hash locked transaction dapat dibuat menggunakan struktur antara lain sebagai berikut, yang merupakan kombinasi antara penggunaan password dengan P2PK.

OP_SHA256 <hash> OP_EQUALVERIFY <pubkey> OP_CHECKSIG

Dengan menggunakan struktur di atas, penerima bitcoin harus juga memperoleh password dari sang pembayar sebelum bisa menggunakan bitcoin yang dibayarkan tersebut. Skema ini dapat digunakan dalam sistem tukar menukar koin yang disebut dengan Atomic Cross Chain Transfer [36].

Transaction Signature

Untuk dapat menggunakan bitcoin yang diterima, penerima harus menandatangan transaksi tersebut dengan menggunakan skema digital signature. Terdapat beberapa langkah yang harus dilakukan untuk membuat digital signature tersebut.

- Buat kopian transaksi aslinya.
- Ganti isi <scriptSig> dengan <scriptPubKey> dari transaksi yang direferensikan oleh transaksi ini. Bagian <scriptSig> akan dibuat setelah proses penandatanganan, sebab bagian ini berisi digital signature.
- Hitung nilai hash dan tandatangan menggunakan kunci privat.

Transaction signature memiliki beberapa tipe hash, yaitu:

- SIGHASH_ALL, tipe hash yang memasukkan semua output, sehingga alamat tujuan tidak dapat diganti lagi.
- SIGHASH_NONE, tipe hash yang tidak memasukkan output, sehingga alamat tujuan dapat diganti.
- SIGHASH_SINGLE, tipe hash yang hanya memasukkan 1 output saja, sedangkan alamat tujuan lain dapat diubah.

Terdapat juga sebuah tipe lain yaitu SIGHASH_ANYONECANPAY yang dapat dikombinasikan dengan salah satu dari tipe-tipe di atas. Tipe ini berarti bahwa "siapapun dapat membayar", di mana input dapat ditambahkan meskipun sudah ditandatangan.

Kontrak

Skema kontrak dapat dibuat dalam sistem Bitcoin dengan memanfaatkan fitur-fitur yang ada [37]. Terdapat beberapa skema kontrak yang dapat dibuat yang tentunya disesuaikan dengan kebutuhan. Fitur-fitur yang dapat digunakan dalam skema kontrak antara lain:

- SIGHASH
- Multisignature
- Tipe Transaksi
- Locktime

Dengan menggunakan multisignature, maka pengguna Bitcoin dapat membuat sistem escrow untuk mengamankan transaksi dari

pembeli atau penjual yang berbuat curang. Sementara dengan menggunakan SIGHASH, pengguna dapat membuat transaksi crowd-funding, yaitu sistem pembiayaan secara bersama-sama. Dengan menggunakan locktime, maka pengguna dapat mengunci dana Bitcoin sampai pada waktu yang telah ditentukan tanpa takut dicuri oleh orang lain.

LockTime

Locktime atau disebut juga nLockTime, merupakan sebuah fitur di dalam sistem Bitcoin yang dapat digunakan untuk menentukan waktu tercepat sebuah transaksi dapat dikonfirmasi ke dalam sistem Bitcoin. Locktime direpresentasikan ke dalam 4 byte data di dalam script transaksi Bitcoin [38]. Locktime dapat digunakan dengan 2 cara. Apabila nilainya kurang dari 500 juta, artinya nilai Locktime menunjukkan nomor blok, sementara apabila nilai Locktime lebih besar daripada 500 juta, maka nilai tersebut menunjukkan jumlah detik setelah 1 Januari 1970 00:00 UTC [8].

Block height atau nomor blok dapat digunakan sebagai timestamp (penanda waktu) karena sistem Bitcoin menciptakan blok baru setiap kira-kira 10 menit. Artinya, jika Locktime disetting 10 blok lebih tinggi dibandingkan nomor blok terbaru, maka dapat dikatakan bahwa transaksi dikunci selama 100 menit. Nilai Locktime dalam script Bitcoin direpresentasikan dalam format low endian. Apabila Locktime bernilai 00000000, itu artinya tidak ada Locktime dan transaksi dapat segera dikonfirmasi.

Sequence Number

Sequence number merupakan informasi sepanjang 4 byte yang dapat digunakan untuk mencatat versi transaksi [39]. Apabila sequence number kurang dari FFFFFFFF (atau 4.294.967.295 dalam bilangan desimal), maka transaksi tersebut dipandang sebagai transaksi yang belum final dan masih dapat diubah. Untuk mengubah transaksi, versi berikutnya dari transaksi arus memiliki sequence number yang lebih besar dibandingkan transaksi sebelumnya.

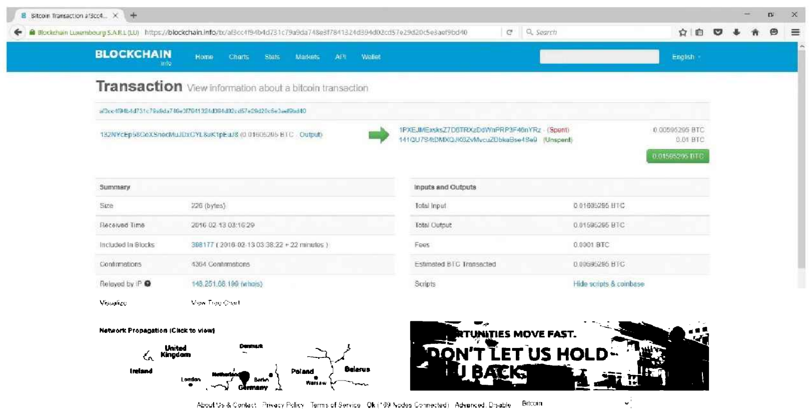
CheckLockTimeVerify

CheckLockTimeVerify (CLTV) merupakan sebuah fitur yang diusulkan oleh Peter Todd [40]. CLTV digunakan untuk mengunci transaksi

sampai dengan waktu yang telah ditentukan. Dengan menggunakan CLTV, sebuah transaksi dapat segera dimasukkan ke dalam blockchain namun tidak dapat digunakan sampai waktu tertentu. CLTV merupakan redefinisi atas perintah OP_NOP2. Informasi CLTV menggunakan tipe data low endian. Selain itu, CLTV juga dapat dikombinasikan dengan LockTime. Jika LockTime digunakan, sequence number juga harus digunakan. Sementara nilai CLTV harus sama dengan atau kurang dari LockTime.

Unspent Transaction

Unspent transaction merupakan transaksi input yang diterima oleh suatu alamat, namun belum dimanfaatkan untuk melakukan pembayaran. Jumlah bitcoin yang dimiliki oleh suatu alamat Bitcoin merupakan penjumlahan dari dana yang diterima dari unspent transaction.



Gambar 13. Contoh Unspent Transaction.

Sebagai contoh pada Gambar 8, transaksi dengan TxID berikut

`af3cc4f94b4d731c79a9da748e3f7841324d394d02cd57e29d-20c5e3aef9bd40`

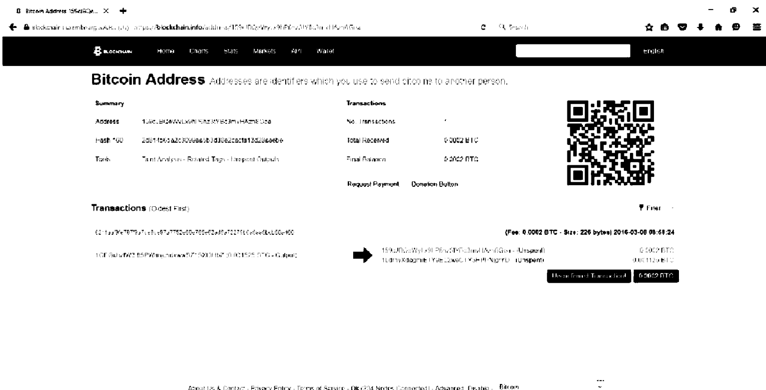
merupakan unspent transaction untuk alamat

`141QU7S4tDMXQJK6ZvMvcuZDbkaBse4Se9`

Unspent transaction dapat digunakan untuk melakukan pembayaran di kemudian hari dari alamat di atas.

Unconfirmed Transaction

Unconfirmed transaction merupakan transaksi yang sudah dikirim ke jaringan Bitcoin namun belum dimasukkan ke dalam blok. Untuk dapat dikonfirmasi ke dalam blok, transaksi-transaksi baru masuk ke dalam antrean.

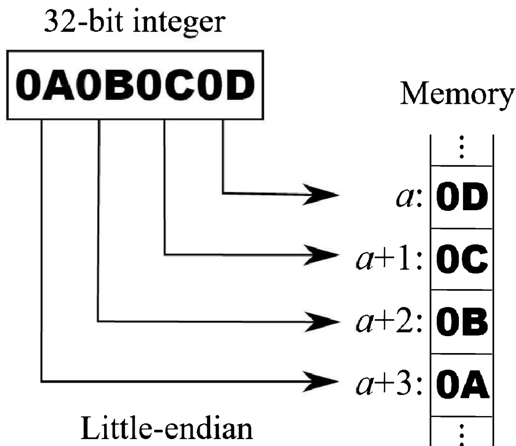


Gambar 14. Unconfirmed transaction.

Antrean unconfirmed transaction diurutkan berdasarkan prioritas. Prioritas tersebut disesuaikan dengan biaya transaksi yang dibayar oleh pengirim bitcoin, artinya semakin besar biaya transaksi, maka semakin besar pula peluang sebuah transaksi tersebut dikonfirmasi dengan cepat.

Little Endian

Little endian merupakan salah satu metode representasi data di mana nilai yang lebih tinggi ditulis mendahului nilai yang lebih rendah [41]. Little endian merupakan kebalikan dari big endian. Sebagai contoh, bilangan 100000 dalam big endian akan ditulis 000010 dalam little endian. Atau sebagai contoh pada Gambar 10, data 0A0B0C0D diubah menjadi 0D0C0B0A.



Gambar 15. Operasi Little Endian di dalam Prosesor [41]

Di dalam Bitcoin, banyak nilai direpresentasikan ke dalam format little endian. Untuk mengubah dari nilai big endian ke little endian, urutan byte ditukar dari kecil ke besar menjadi dari besar ke kecil.

Terdapat perdebatan mengapa Bitcoin harus menggunakan little endian. Beberapa orang mengatakan bahwa penggunaan little endian dilakukan untuk optimisasi, sehingga akan meningkatkan kecepatan perhitungan di level prosesor [42]. Beberapa juga berpendapat bahwa little endian digunakan untuk optimasi jaringan komputer [43].

Namun ada pula pihak-pihak yang mengatakan bahwa optimisasi little endian tidak diperlukan karena prosesor modern dapat melakukan operasi reverse (pembalikan) urutan byte dengan mudah. Bagaimanapun alasannya, Bitcoin hingga saat ini tetap menggunakan little endian, dan meskipun cukup mempersulit pemahaman mereka yang sedang mempelajari sisi teknis Bitcoin, para pengembang Bitcoin Core tidak berencana untuk mengubah little endian menjadi big endian.

Biaya Transaksi

Biaya transaksi dalam Bitcoin merupakan sejumlah bitcoin yang dibayarkan oleh pengirim uang kepada para penambang sebagai upah atas usahanya menambang. Biaya transaksi ini merupakan salah satu dari 2 sumber pemasukan penambang Bitcoin selain mining reward (hadiah penambangan).

Terdapat rumus yang dapat digunakan untuk menghitung ukuran transaksi standar (transaction size atau TxSize) yang dihitung dalam satuan byte adalah sebagai berikut [44].

TxSize = 148 x Jumlah Input + (34 x Jumlah Output) + 10

Setelah mendapatkan ukuran transaksi, maka dapat diitung biaya transaksi (transaction fee atau TxFee) dalam satuan satoshi sebagai berikut [45].

TxFee = (TxSize / 1000) x 10000

Menurut rumus di atas, biaya transaksi adalah sebesar 10.000 satoshi untuk setiap 1.000 byte atau 1 KB. Sesuai kesepakatan pula, biaya transaksi minimum saat ini adalah sebesar 10.000 satoshi [46], artinya meskipun ukuran transaksi kurang dari 1 KB, namun biaya transaksi harus tetap 10.000 satoshi. Kemudian, jika ukuran transaksi sebesar 1.001 byte sampai dengan 2.000 byte, maka biaya transaksi yang dibayarkan minimal sebesar 20.000 satoshi, demikian berlaku kelipatannya.

Satuan Unit Bitcoin

Terdapat banyak satuan unit Bitcoin yang dapat digunakan [47]. Beberapa di antara satuan unit yang sering digunakan adalah sebagai berikut.

1 bitcoin (BTC)	=	10 deci-bitcoin (dBTC)
	=	100 centi-bitcoin (cBTC)
	=	1.000 milli-bitcoin (mBTC)
	=	1.000.000 micro-bitcoin (μBTC)
	=	100.000.000 satoshi (sat)

Selain menggunakan bilangan bulat atau dengan menggunakan satuan yang lebih kecil, di dalam penggunaan sehari-hari, 1 dBTC juga biasa disebut dengan 0,1 BTC, dan 1 cBTC menjadi 0,01 BTC, dan 10.000 satoshi juga biasa ditulis 0,0001 BTC.

Wallet Bitcoin

Untuk dapat menggunakan sistem Bitcoin dengan mudah, pengguna harus menggunakan perangkat lunak yang disebut wallet Bitcoin. Meskipun disebut dengan wallet, namun tidak ada bitcoin yang disimpan dalam wallet tersebut, melainkan hanya membantu pengguna untuk mengorganisasikan alamat Bitcoin dan melindungi kunci privat yang digunakan untuk menandatangan transaksi. Terdapat beberapa fitur dasar dari wallet yang dapat dimanfaatkan oleh pengguna. Pertama, wallet mengirim permintaan informasi kepada blockchain untuk menghitung total saldo bitcoin yang dimiliki oleh pengguna tersebut. Kemudian, wallet juga dapat digunakan untuk membuat alamat Bitcoin baru yang dapat digunakan dalam transaksi-transaksi baru. Wallet terutama versi mobile juga memiliki fitur untuk membaca QR code dengan memanfaatkan fitur kamera pada smartphone. Fitur ini sangat penting sebab alamat Bitcoin tidak mudah dihafalkan dan QR code menjadi standar dalam bertukar informasi alamat Bitcoin.

Software Wallet

Terdapat berbagai macam jenis perangkat lunak wallet, yang tersedia dalam berbagai platform seperti web, desktop, dan mobile. Wallet juga memiliki berbagai macam fungsi, di antaranya wallet yang hanya digunakan untuk menerima bitcoin namun tidak dapat digunakan untuk mengirim bitcoin. Wallet jenis ini biasanya digunakan pada server toko atau penjual untuk meningkatkan keamanan, sebab kunci privat diamankan di tempat lain seandainya server tersebut berhasil ditembus oleh pencuri data [9].

Deterministic Wallet

Deterministic wallet merupakan jenis wallet yang dapat menciptakan alamat baru berdasarkan skema tertentu sehingga alamat baru tersebut merupakan turunan dari sebuah alamat utama. Karena merupakan turunan, alamat-alamat baru tersebut tidak perlu disimpan karena dapat dihitung ulang selama data kunci privat dari alamat utama masih tersedia [9]. Dengan demikian, deterministic wallet lebih mudah dibackup [48].

Terdapat 2 tipe deterministic wallet, yakni tipe-1 dan tipe-2. Tipe-1 membuat alamat Bitcoin baru dari sebuah password yang digabungkan dengan sebuah angka counter. Private key dihitung dengan

menghitung nilai hash dari kombinasi antara password dan counter, oleh karena itu private key tidak perlu disimpan karena dapat dihitung kapan saja asal nilai password dan counter dapat diketahui.

Proses pembuatan private key dan public key dalam deterministic wallet tipe-1 dapat ditampilkan dengan menggunakan rumus sebagai berikut.

$$\mathbf{priv=H(pw|n)}$$
$$\mathbf{B=priv\cdot A \bmod p}$$

dengan priv adalah private key, pw adalah password, dan n adalah bilangan counter, sedangkan H adalah fungsi hash. B merupakan public key yang dihitung dengan menggunakan private key priv, A, dan p.

Deterministic wallet tipe-2 dapat dijelaskan sebagai berikut. Proses pembuatan public key B dapat dilakukan dengan menggunakan rumus:

$$\mathbf{priv=mpk+H(pw|n)}$$
$$\mathbf{B=priv \cdot A \bmod p}$$
$$\mathbf{B=mpk\cdot A+H(pw|n)\cdot A \bmod p}$$
$$\mathbf{B=B_mpk+H(pw|n)\cdot A \bmod p}$$

dengan mpk sebagai master private key, B_{mpk} sebagai master public key. Tipe-2 ini memisah antara master private key mpk dan password, dan dengan demikian dapat meningkatkan keamanan dan kemanfaatan skema ini. Dengan menggunakan skema ini, pengguna dapat dengan aman membuat wallet yang hanya dapat menerima dana dengan menggunakan password pw dan master public key B_{mpk} tanpa harus menyebutkan private key mpk.

Hierarchical Deterministic Wallet

Hierarchical Deterministic wallet (HD wallet) merupakan tipe deterministic wallet dengan kemampuan membuat alamat baru dari sebuah kunci parent atau induk (baik public key maupun private key) [49]. Proses konstruksi child address dari parent address berbentuk seperti pohon hirarki. Sementara itu, public key dan private key dapat diturunkan secara terpisah. Setiap public key dapat digunakan untuk

membuat 231 child public key baru [9]. Child key dapat dibuat dari parent key, tetapi tidak sebaliknya, parent key tidak dapat dibuat dari child key.

Parent Private Key
Parent Chain Code
Parent Public Key
Index Number
One-Way Hash
Child Private Key
Child Chain Code
Child Public Key
Mathematical Relationship
Derived Mathematical Relationship

Gambar 16. Proses pembuatan key dalam HD wallet

Dark Wallet

Dark wallet1 merupakan software wallet Bitcoin yang diciptakan oleh Cody Wilson dan Amir Taaki sebagai sebuah usaha untuk menciptakan transaksi anonim di dalam sistem Bitcoin [50]. Dark wallet menggunakan sebuah teknik yang disebut stealth address yang diciptakan oleh Peter Todd [51], salah satu pengembang Bitcoin Core yang ternama. Saat buku ini ditulis, versi terakhir Dark Wallet yang tersedia adalah Alpha 8 dan tersedia sebagi ekstensi peramban Google Chrome.

Wiki Community Presskit Donate
Darkwallet
Introducing the digital wallet for bitcoin economics.
Darkwallet is an open source bitcoin wallet that provides a solid foundation of well designed asynchronous code for scalability.

Gambar 17. Situs Dark Wallet

1 http://darkwallet.is

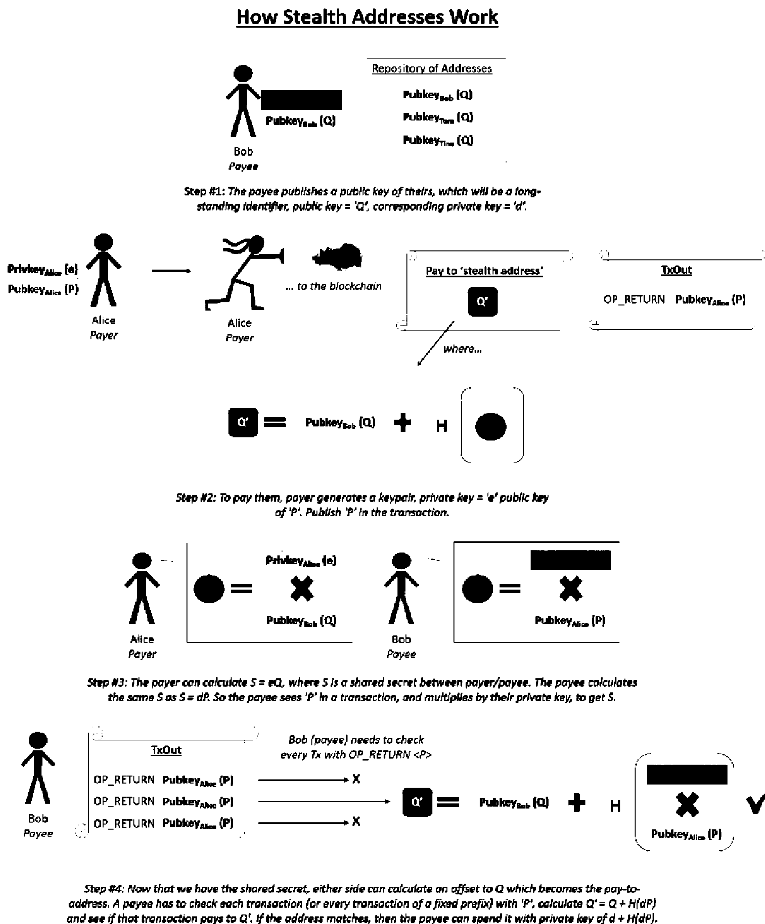
Meskipun Dark Wallet masih dalam tahap pengembangan, software ini menawarkan skema anonim untuk mengamankan transaksi Bitcoin. Alamat stealth akan dibuat oleh wallet ini, sehingga pemilik alamat stealth tidak perlu memberitahukan alamat Bitcoin kepada siapapun yang hendak mengirimkan uang kepadanya.

Alamat stealth menggunakan skema Elliptic Curve Diffie Hellman (ECDH) untuk bertukar informasi. Langkah-langkah pembuatan alamat stealth dapat dijelaskan sebagai berikut [52].

- Penerima mempersiapkan Q = dG, di mana Q adalah public key EC, d adalah private key, dan G adalah generator EC.
- Pembayar mempersiapkan P = eG, di mana e adalah private key.
- P dipublikasikan. Dengan demikian, penerima dan pembayar dapat menghitung c = H(eQ) = H(dP), dengan H adalah fungsi kriptografi tertentu.
- Pembayar menciptakan public key baru Q' = Q + cG
- Penerima dapat menghitung Q = Q + cG = (d + C) G dan private key e = (d + c)

Karena pembayar tidak perlu berkomunikasi dengan penerima terkait dengan pembayaran yang dilakukan, maka penerima dapat mencari transaksi tersebut di dalam blockchain atas transaksi yang memiliki informasi OP_RETURN tertentu. Apabila transaksi ditemukan, maka alamat tersebut dimasukkan ke dalam dark wallet. Selain menggunakan stealth address, dark wallet juga tidak menggunakan alamat yang pernah dipakai. Untuk setiap transaksi baru akan diciptakan alamat baru pula.

Proses transaksi menggunakan dark wallet dapat dijelaskan dalam Gambar 14.



Gambar 18. Cara Kerja Stealth Address [52]

OpCode

OpCode atau Operation Code merupakan kode-kode yang digunakan dalam script transaksi Bitcoin [8]. OpCode direpresentasikan ke dalam kode angka heksadesimal. OpCode digunakan untuk memberi informasi kepada sistem dalam mengolah informasi di dalam script.

Di dalam Bitcoin tidak dikenal perintah repetisi atau perulangan, oleh sebab itu sistem Bitcoin disebut sebagai sistem yang "non-Turing complete". Hal ini sengaja dilakukan untuk menghemat sumber daya komputasi di dalam jaringan Bitcoin dan menghindari serangan penjahat yang berusaha mengganggu jalannya sistem Bitcoin. Daftar lengkap beserta penjelasannya akan disediakan di dalam lampiran.

Menyusun Bitcoin Script

Bitcoin mengevaluasi script dalam sistem stack yang mirip seperti bahasa pemrograman Forth [30]. Dengan adanya sistem stack, maka informasi diolah dengan cara Last In First Out (LIFO). Script terkustomisasi Bitcoin umumnya digunakan dalam skema pembayaran Pay To Script Hash (P2SH).

Untuk dapat menyusun Bitcoin script, maka ada beberapa hal yang perlu dilakukan.

Memahami OpCode.

Sebelum mulai menyusun script, ada baiknya memahami bentuk-bentuk OpCode yang didukung oleh Bitcoin agar dapat melihat kemampuan dan batasan yang ada di dalam sistem Bitcoin. Sebagai contoh, Bitcoin hanya memiliki beberapa jenis fungsi hash, tidak memiliki fungsi iterasi atau perulangan, dan juga memiliki batasan jumlah data yang dapat diolah.

Menentukan tujuan.

Sebelum memulai pembuatan script, ada baiknya menentukan tujuan kustomisasi script Bitcoin. Apabila script standar sudah dapat memenuhi keperluan, maka script kustom tidak perlu dibuat. Namun jika memerlukan modifikasi atas script standar, maka script kustom dibuat dengan memperhatikan tujuan pembuatan.

Sebagai contoh, script kustom yang digunakan untuk mengunci transaksi dengan menggunakan sebuah password, maka script ini membutuhkan beberapa jenis operasi seperti operasi hash dan operasi validasi digital signature.

Menyusun algoritma.

Algoritma script dapat dibuat dengan menggunakan pseudocode. Perlu diketahui pula bahwa OpCode di dalam Bitcoin tidak persis sama penggunaannya dengan kode-kode pemrograman.

Sebagai contoh, berikut algoritma atas script untuk melakukan pembayaran dengan syarat pihak penerima memiliki password tertentu, jika sampai batas waktu tertentu pihak penerima tidak dapat menunjukkan password tersebut, maka pihak pembayar berhak menarik kembali pembayaran.

Pseudocode:

```
If
    (password) and (payee's signature) are correct
    then pay
else
    if (time expired)
    then if (payer's signature) is correct
         then refund
endif;
```

Catatan: pseudocode di atas ditulis dalam bahasa Inggris standar. Payee merupakan pihak penerima pembayaran, sementara Payer maksudnya pihak pembayar.

Menyusun script.

Setelah rancangan script berhasil disusun, maka langkah berikutnya adalah mengubahnya menjadi OpCode. Sebagai contoh, dari langkah (c) di atas akan diperoleh script sebagai berikut dengan memperhatikan OpCode dan syarat-syarat yang ditentukan untuk masing-masing OpCode yang digunakan.

```
OP_IF
    OP_SHA256 <secret key> OP_EQUALVERIFY
    <Payee's pubkey> OP_CHECKSIG
OP_ELSE
    <expiry time> OP_CHECKLOCKTIMEVERIFY OP_DROP
    <Payer's pubkey> OP_CHECKSIG
OP_ENDIF
```

Melakukan pengujian.

Pengujian dilakukan untuk memastikan bahwa script dapat dieksekusi dengan sempurna. Untuk mempermudah pengujian, dapat

digunakan fasilitas milik Webbtc.com melalui https://webbtc.com/script.

Setelah sukses diuji melalui laman di atas, perlu juga dilakukan pengujian melalui script Testnet3 maupun Mainnet. Meskipun kedua jaringan ini seharusnya sama, namun pada kenyataannya tidak semua script dapat dijalankan dengan baik pada Testnet3, sebab Testnet3 tidak dikelola dengan baik, di antaranya perangkat lunak node-node dalam Testnet3 tidak terupdate dengan baik, dan juga adanya penambang yang menguasai lebih dari 50% kekuatan komputasi di dalam Testnet juga mempengaruhi pengujian.

Berikut ini adalah contoh transaksi P2SH yang menggunakan script kustom di atas.

Redeem TxID: `e348f2f44883ceae6112b36d85ced917333be49a-c81aad79d37b177ba31b93f8`

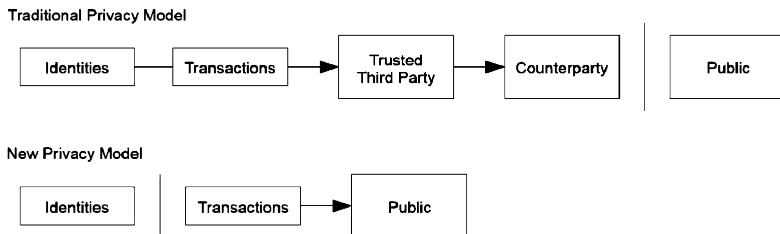
Commit TxID: `27946ebf044a4e4c611685fa9e1ad71c-87107da8364df001764de70483c39de4`

BAB 3
Privasi dan Anonimitas

Privasi merupakan masalah besar dalam sistem Bitcoin. Meskipun pada dasarnya Bitcoin menawarkan transaksi anonim, beberapa teknik telah dikembangkan untuk membuka hubungan antara alamat Bitcoin, pola-pola transaksi, dan identitas asli pemilik alamat Bitcoin tersebut.

Masalah Privasi

Bitcoin didesain dengan model privasi di mana transaksi yang dilakukan dan alamat yang dimiliki oleh seorang pengguna tidak memiliki keterkaitan langsung dengan identitas asli pemiliknya. Model privasi Bitcoin dan perbandingannya dengan model privasi tradisional dapat dideskripsikan pada Gambar 15.



Gambar 19. Model Privasi Bitcoin [1]

Semua orang dapat bergabung ke dalam sistem Bitcoin tanpa harus mendaftar dahulu, sebab tidak ada organisasi pengendali terpusat di dalam sistem Bitcoin yang mengendalikan pengguna maupun transaksi yang terjadi di dalam sistem. Meskipun transaksi-transaksi tersebut dapat dilihat oleh semua orang, identitas yang terkait dengan transaksi tetap tersembunyi.

Namun demikian, model privasi seperti ini tidak berarti identitas pengguna tetap tersembunyi. Terdapat banyak peraturan yang ditetapkan dan juga karakteristik Bitcoin yang dapat digunakan untuk menganalisis hubungan antara transaksi-transaksi bitcoin dan identitas asli pengguna. Hal inilah yang kemudian menyebabkan komunitas Bitcoin cenderung menyebut Bitcoin sebagai pseudo-anonym atau anonimitas semu.

Prinsip KYC

Pemerintah-pemerintah di dunia mulai sadar akan skema pencucian uang yang dapat dilakukan dengan menggunakan mata uang digital seperti Liberty Reserve [53], yang kemudian memaksa insitusi keuangan untuk menerapkan prinsip Know Your Customer (KYC). Dengan menerapkan prinsip KYC, tidak ada orang yang bisa membuat akun bank tanpa kartu identitas. Prinsip yang sama dipaksakan

kepada institusi keuangan yang terkait dengan sistem Bitcoin, seperti perusahaan perdagangan Bitcoin yang memperbolehkan pengguna untuk menjual maupun membeli bitcoin dan mengkonversikan mata uang lokal ke dalam bitcoin atau sebaliknya [54].

Layanan jual beli bitcoin di Amerika Serikat harus mendaftarkan diri sebagai entitas bisnis jasa keuangan, dan oleh karena itu harus mengikuti aturan-aturan yang sama seperti layanan keuangan tradisional lainnya seperti perbankan [55]. Sementara itu, layanan jual beli di luar Amerika Serikat seperti Mt. Gox di Jepang (sudah bangkrut), CoinJar di Australia, dan Bitcoin.co.id di Indonesia mengadopsi mekanisme yang sama dengan mempersyaratkan para pengguna mereka untuk menyerahkan salinan identitas diri untuk divalidasi secara manual oleh perusahaan-perusahaan tersebut. Artinya, layanan jual beli bitcoin memiliki catatan koneksi antara alamat bitcoin dengan identitas asli penggunanya. Karena setiap transaksi di dalam sistem Bitcoin dapat dilihat oleh siapapun, bitcoin yang dibeli dari layanan jual beli bitcoin dapat dilacak dengan mudah. Oleh karena itu, apabila transaksi tersebut dicurigai terkait aktivitas ilegal, maka identitas pengguna yang terkait dapat diidentifikasi jika penggunanya membeli maupun menjual bitcoin mereka langsung dari layanan jual beli bitcoin.

Greenlist

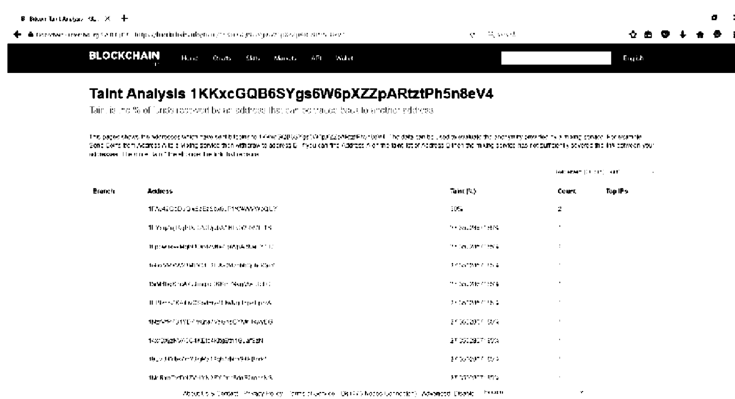
Usaha lain untuk yang dapat digunakan untuk memenuhi peraturan yang ditetapkan oleh pemerintah Amerika Serikat adalah dengan menggunakan skema greenlist [56]. Greenlist adalah skema yang merekam informasi identitas asli pemilik alamat Bitcoin. Konsep ini diajukan pertama kali oleh perusahaan CoinValidation yang memegang informasi tersebut. Dengan skema Greenlist, perusahaan akan melakukan verifikasi kepemilikan alamat Bitcoin. Dengan demikian, informasi ini dapat digunakan oleh penegak hukum untuk melacak mereka yang membeli barang-barang terlarang dengan menggunakan bitcoin [57]. Ide Greenlist sejalan dengan kebijakan Gedung Putih yang disebut National Strategy for Trusted Identities in Cyberspace [58] yang telah ditandatangani Presiden Barack Obama.

Ide Greenlist telah menerima penolakan dari komunitas Bitcoin yang diekspresikan dalam forum Bitcointalk [59] dan Reddit [60], di mana para pengguna forum telah menyampaikan bahwa Greenlist akan menghilangkan konsep fungibilitas, yakni suatu konsep di mana

bitcoin manapun dengan jumlah yang sama seharusnya memiliki nilai yang sama pula terlepas dari mana bitcoin tersebut berasal: bitcoin baru harusnya punya nilai yang sama dengan bitcoin yang dimiliki oleh pedagang narkoba. Skema Greenlist akan menciptakan valuasi bitcoin yang berbeda-beda tergantung apakah bitcoin tersebut berasal dari alamat yang telah teregistrasi. Kebijakan Greenlist dapat menjadi titik awal bagi agen pemerintah yang memiliki akses basis data untuk melacak identitas siapapun yang bertransaksi dengan alamat-alamat Bitcoin tertentu [56].

Taint

Taint merupakan jejak transaksi yang dapat digunakan untuk mengukur konektivitas antara sebuah alamat Bitcoin dengan alamat Bitcoin lain yang terhubung dengan transaksi di antara alamat-alamat tersebut [61]. Alamat-alamat yang saling bertransaksi cenderung memiliki keterkaitan akan sesuatu, misalnya dimiliki oleh pengguna yang sama, atau memiliki hubungan penjual-pembeli dalam skema transaksi jual beli barang maupun jasa.



Gambar 20. Taint Analysis dari Blockchain.info

Gambar 16 menunjukkan fasilitas Taint Analysis yang disediakan oleh situs Blockchain.info. Dari Taint Analysis tersebut, semakin besar persentase taint, maka semakin dekat keterkaitan di antara alamat-alamat tersebut.

Solusi Anonimitas Bitcoin

Terdapat berbagai solusi untuk memecahkan permasalahan anonimitas semu Bitcoin yang ditawarkan berbagai pihak. Solusi-solusi tersebut akan dibahas di dalam subbab ini. Beberapa solusi anonimitas sudah ditawarkan sebagai layanan, namun banyak juga solusi yang masih dalam tahap konseptual.

Tor

Tor merupakan sebuah jaringan yang memungkinkan penggunanya untuk memiliki komunikasi anonim dengan mengenkripsi seluruh data yang ada di dalam jaringan dan menciptakan jalur-jalur acak di dalam jaringan yang terususun atas banyak server relay [62]. Tor merupakan singkatan dari "The Onion Ring", nama awal dari sebuah proyek yang diinisiasi oleh Angkatan Laut Amerika Serikat.

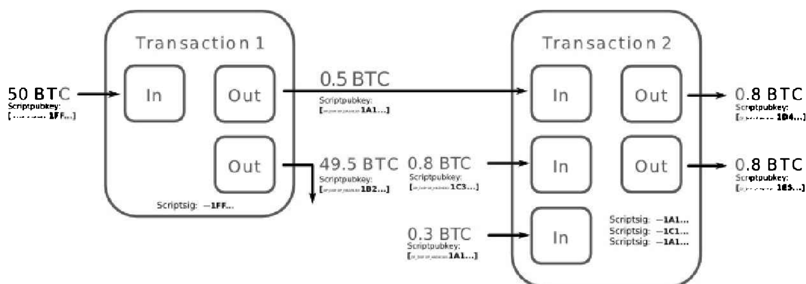
Tor dapat digunakan untuk menghindari pantauan jaringan dan analisis lalu-lintas jaringan komputer, karena semua komunikasi pengguna dienkripsi dan melalui jalur acak dalam server-server relay dan menciptakan sebuah sirkuit setiap kali terhubung dengan server [63]. Tor sering digunakan oleh pengguna Bitcoin untuk menambahkan aspek privasi karena kemampuannya menyembunyikan informasi transaksi di dalam jaringan [64]. Hal ini disebabkan karena jaringan Bitcoin sendiri tidak mengenkripsi paket data transaksi, sehingga pihak penyedia layanan internet (Internet Service Provider atau ISP) dapat memantau transaksi bitcoin yang terjadi di dalam jaringan miliknya.

Meskipun ide menjalankan sistem bitcoin di dalam jaringan Tor tampaknya menjadi hal yang menjanjikan privasi, namun masih ada kemungkinan bahwa komunikasi di dalam Tor dapat diketahui oleh pihak lain, dengan menggunakan mekanisme tertentu [65]. Dan Kaminsky juga memaparkan salah satu cara untuk memantau transaksi Bitcoin yang dilakukan dalam jaringan Tor dengan memanfaatkan karakteristik peer-to-peer. [66]

CoinJoin

Gregory Maxwell memperkenalkan solusi alternatif untuk meningkatkan level privasi pengguna Bitcoin yang disebut dengan CoinJoin [67] yang merupakan pengembangan atas ide yang disampaikannya sebelumnya tentang taint [68]. CoinJoin merupakan mekanisme yang

menggabungkan beberapa transaksi serupa ke dalam sebuah transaksi yang terdiri atas banyak input dan banyak output. Konsep CoinJoin kemudian diimplementasikan ke dalam sebuah aplikasi bernama sama, CoinJoin [69].



Gambar 21. Transaksi CoinJoin [67]

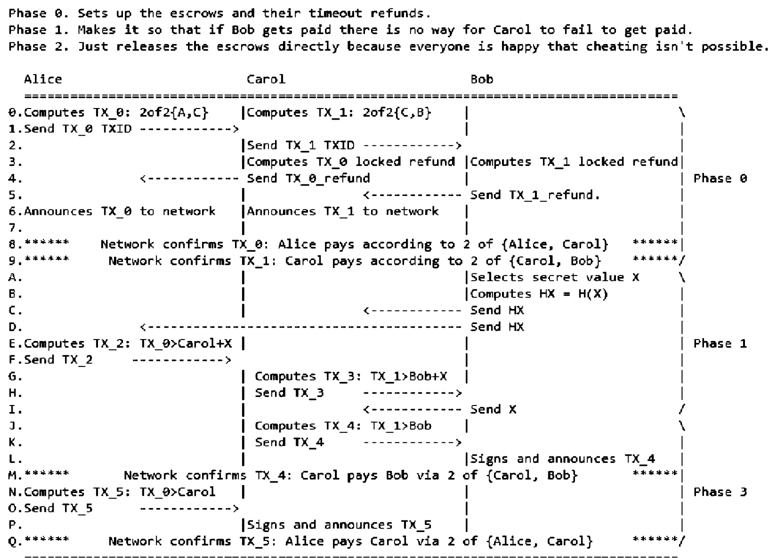
Gambar 17 menjelaskan cara kerja CoinJoin dan perbandingannya dengan transaksi Bitcoin. Pada transaksi 1 (transaction 1), sebuah alamat 1FF yang memiliki 50 BTC ingin mengirim 0,5 BTC ke alamat lain, 1A1 dan dengan alamat kembalian 1FF. Pada akhir transaksi nanti, alamat 1FF akan memiliki bitcoin sebanyak 49.5 BTC.

Pada transaksi 2 (transaction 2), terdapat banyak alamat input dan alamat output. Meskipun skema ini tampak seperti transaksi Bitcoin biasa, namun dapat digunakan untuk melindungi identitas pengguna yang terkait dengan transaksi terseut. Misalnya pemilik alamat 1A1 ingin mengirim 0,8 BTC kepada 1E5 dan tidak ingin orang lain mengetahui transaksi ini, maka ia mengkombinasikan transaksinya dengan transaksi lain dengan ukuran yang sama, misalnya dari alamat 1C3 ke 1D4. Artinya dengan melihat pada transaksi, pengamat tidak dapat menentukan alamat mana yang menerima alamat 1A1, karena bisa saja bitcoin tersebut berasal dari 1D4 atau 1E5.

CoinSwap

Ide CoinSwap pertama kali diciptakan oleh Gregory Maxwell [70] di forum Bitcointalk. CoinSwap akan melindungi informasi transaksi antara pembayar dan penerima. CoinSwap memungkinkan pihak-pihak yang terlibat untuk membuat transaksi dengan melibatkan garansi yang membuat tidak ada pihak yang dapat mencuri bitcoin dari pihak

lain. Dalam CoinSwap, pihak ketiga dibutuhkan sebagai penghubung antara pihak pembayar dan penerima, dan juga beberapa transaksi harus dibuat dalam skema CoinSwap. Beberapa metode transaksi yang digunakan dalam CoinSwap di antaranya 2-of-2 escrow sebagaimana dibahas pada bab 2.2.7 dan hash-locked transaction pada bab 2.2.8.



```
Phase 0. Sets up the escrows and their timeout refunds.
Phase 1. Makes it so that if Bob gets paid there is no way for Carol to fail to get paid.
Phase 2. Just releases the escrows directly because everyone is happy that cheating isn't possible.

  Alice                        Carol                        Bob
  ==================================================================================
0.Computes TX_0: 2of2{A,C}    |Computes TX_1: 2of2{C,B}    |                         \
1.Send TX_0 TXID ------------>                             |                         |
2.                            |Send TX_1 TXID ------------>                          |
3.                            |Computes TX_0 locked refund |Computes TX_1 locked refund|
4.             <------------ Send TX_0_refund              |                         | Phase 0
5.                            |             <------------ Send TX_1_refund.          |
6.Announces TX_0 to network   |Announces TX_1 to network   |                         |
7.                            |                            |                         |
8.******    Network confirms TX_0: Alice pays according to 2 of {Alice, Carol}   ******|
9.******     Network confirms TX_1: Carol pays according to 2 of {Carol, Bob}    ******/
A.                            |                            |Selects secret value X    \
B.                            |                            |Computes HX = H(X)        |
C.                            |             <------------ Send HX                     |
D.             <----------------------------------------- Send HX                     |
E.Computes TX_2: TX_0>Carol+X |                            |                         | Phase 1
F.Send TX_2      ------------>                             |                         |
G.                            | Computes TX_3: TX_1>Bob+X  |                         |
H.                            | Send TX_3     ------------>                          |
I.                            |             <------------ Send X                     /
J.                            | Computes TX_4: TX_1>Bob    |                         \
K.                            | Send TX_4     ------------>                          |
L.                            |                            |Signs and announces TX_4 |
M.******        Network confirms TX_4: Carol pays Bob via 2 of {Carol, Bob}       ******|
N.Computes TX_5: TX_0>Carol   |                            |                         | Phase 3
O.Send TX_5      ------------>                             |                         |
P.                            |Signs and announces TX_5    |                         |
Q.******     Network confirms TX_5: Alice pays Carol via 2 of {Alice, Carol}     ******/
  ==================================================================================
```

Gambar 22. Protokol CoinSwap [70].

Protokol CoinSwap dapat dijelaskan sebagai berikut. Alice bertindak sebagai pembayar, Bob sebagai penerima pembayaran, dan Carol sebagai pihak perantara. Protokol ini dibagi menjadi 3 fase, yakni fase 0, fase 1, dan fase 2.

Dalam fase 0 terdapat proses pembuatan transaksi escrow di antara peserta-peserta tersebut. Alice membuat transaksi escrow TX_0 yang merupakan 2-of-2 escrow antara Alice dan Carol dengan menggunakan bitcoin milik Alice. Transaksi ini dapat dibaca sebagai "Alice akan membayar sejumlah bitcoin kepada Carol jika Alice dan Carol setuju". Kemudian, Carol juga membuat transaksi TX_1 yang merupakan 2-of-2 escrow antara Carol dan ob dengan menggunakan bitcoin milik Carol. Transaksi ini dapat dibaca sebagai "Carol akan membayar sejumlah bitcoin kepada Bob jika Carol dan Bob setuju".

Untuk mengamankan fase ini, Carol membuat TX_0_refund den-

gan batas waktu tertentu yang akan memuat Alice memperoleh kembali bitcoin yang dibayarkan kepada Carol jika sesuatu terjadi di masa depan. Sama seperti Alice, Carol juga membuat TX_1_refund yang akan membuat Carol memperoleh kembali bitcoin yang dibayarkan kepada Bob jika sesuatu terjadi di masa depan. Transaksi TX_0_refund dan TX_1_refund tidak dikirim ke jaringan oleh Alice maupun Carol. Transaksi yang dikirim ke jaringan Bitcoin pada tahap ini adalah TX_0 dan TX_1.

Fase 1 mempersiapkan garansi transaksi untuk semua partisipan yang terlibat. Artinya jika Bob mengambil koin dari Carol, Carol juga dapat mengambil pembayaran dari Alice. Skema ini didukung oleh skema hash-locked transaction. Dalam skema ini, Bob memilih kata kunci rahasia X, kemudian menghitung nilai HX yang merupakan nilai hash dari X dengan rumus HX = H(X) dan mengirim HX kepada Alice dan Carol. Alice membuat transaksi baru TX_2 yang akan mengambil koin dari TX_0 denagn menggunakan tanda tangan Carol dan nilai X dan membayarkannya kepada alamat Carol. Sama seperti Alice, Carol membuat transaksi baru dengan skema hash-locked transaction TX_3 yang akan mengambil koin dari TX_1 dengan menggunakan tanda tangan Bob dan nilai X dan membayarkannya kepada Bob. Pada fase ini, jika Bob curang dan mengambil koin dengan menggunakan TX_3, maka Carol dapat mengambil pembayaran dengan menggunakan TX_2 karena nilai X dapat diketahui ketika Bob menggunakan TX_3 dan mempublikasikannya ke jaringan Bitcoin.

Apabila TX_2 dan TX_3 tidak dikirimkan ke jaringan Bitcoin, maka fase 2 dapat dilanjutkan. Dalam fase ini, Bob memberitahukan nilai X kepada Carol, dan Carol membuat transaksi baru TX_4 yang mengambil koin dari TX_1 dengan menggunakan tanda tangan Bob. Bob dapat menggunakan transaksi TX_4 untuk mendapatkan pembayaran. Sama seperti itu, Alice membuat transaksi TX_5 yang akan mengambil koin dari TX_0 dengan menggunakan tanda tangan Carol dan dibayarkan kepada Carol. Carol dapat mengirimkan transaksi TX_5 ke jaringan Bitcoin untuk mendapatkan pembayaran.

Apabila protokol ini berhasil diselesaikan, maka hanya transaksi TX_4 dan TX_5 yang menggunakan TX_0 dan TX_1 saja yang dikirim ke jaringan Bitcoin, sementara transaksi-transaksi lain dihapus dan tidak dikirimkan ke jaringan Bitcoin.

CoinSwap merupakan salah satu contoh dari metode Zero Knowl-

edge Contingent Payment [4] yang memiliki fitur-fitur sebagai berikut:

- Hash-locked transaction
- Time-locked transaction
- 2-of-2 escrow transaction

Di dalam CoinSwap, setiap langkah dari protokol harus diikuti dengan urutan yang benar agar menghasilkan transaksi yang aman dan bergaransi tanpa adanya pihak terpercaya.

Layanan Mixing

Terdapat metode lain untuk menangani masalah privasi dalam transaksi Bitcoin dengan menggunakan layanan mixing (pengacakan) seperti OnionBC , Bitcoin Fog, BitLaundry, dan Send Shared [54]. Send Shared sendiri telah bertransformasi menjadi Shared Coin. Layanan-layanan ini memiliki metode yang berbeda-beda dalam mencampur bitcoin milik pengguna. Meski demikian, metode yang mereka gunakan dapat diklasifikasikan ke dalam 2 kelompok.

Kelompok pertama, layanan tersebut meminta pengguna mengirim bitcoin ke dalam wallet virtual yang dikendalikan oleh layanan, sehingga pengguna dapat mengambil kembali bitcoin mereka untuk dikirimkan ke alamat Bitcoin lain. Penyedia layanan akan menukar bitcoin pengguna dengan bitcoin lain yang tidak memiliki hubungan dengan bitcoin miliknya sebelumnya. Apabila pengguna ingin membayar koin kepada pihak lain, mereka dapat memasukkan alamat tujuan tersebut untuk dibayarkan oleh sang penyedia layanan. Layanan yang termasuk dalam kelompok ini di antaranya OnionBC, Bitcoin Fog, dan BitLaundry.

Kelompok kedua merupakan layanan yang menggabung-gabungkan beberapa transaksi ke dalam sebuah transaksi besar dengan menggunakan konsep seperti milik CoinJoin. Contoh dari layanan ini adalah Send Shared.

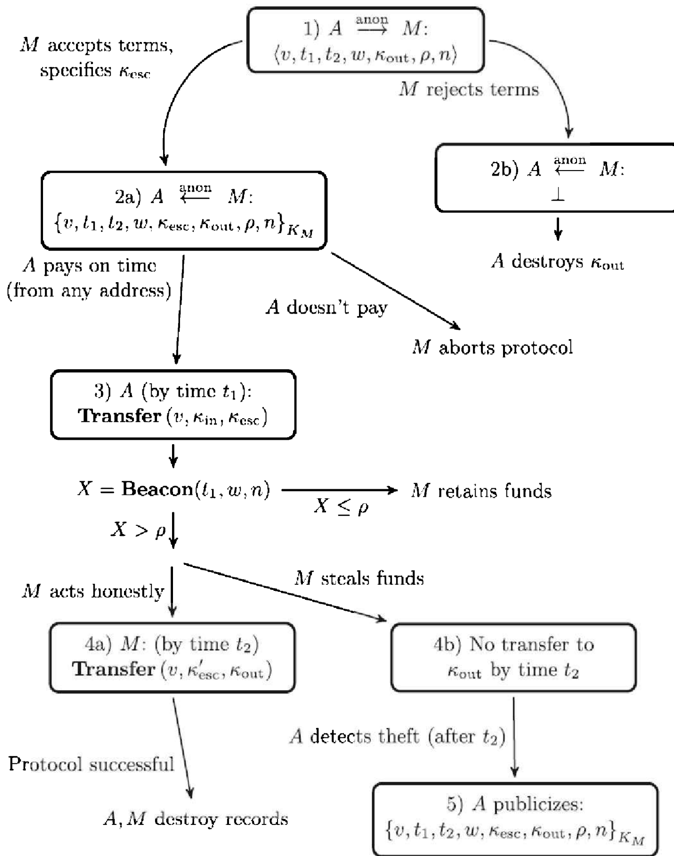
Ketika pengguna menggunakan layanan seperti ini, mereka harus membayar biaya layanan berupa bitcoin kepada pengelola layanan dengan jumlah antara 0,5% sampai dengan 3%. Meskipun layanan seperti ini mungkin dapat meningkatkan anonimitas, namun terdapat pula risiko yang ditanggung oleh pengguna, sebab mereka tidak dapat mengendalikan bitcoin milik mereka ketika mereka telah men-

girimkan bitcoin tersebut kepada alamat Bitcoin milik pengelola. Artinya ketika pihak pengelola layanan bertindak curang dengan mencuri bitcoin milik pengguna, pengguna tidak dapat berbuat apa-apa lagi karena seluruh transaksi Bitcoin tidak dapat dibatalkan.

MixCoin

MixCoin merupakan sebuah konsep yang menciptakan akuntabilitas bagi layanan mixing [71]. Implementasi MixCoin tidak membutuhkan perubahan protokol Bitcoin, sehingga dapat diimplementasikan dengan mudah oleh pengguna. Dalam MixCoin, terdapat 2 pihak yang terkait. Pihak pertama merupakan pihak yang ingin melakukan pengacakan bitcoin, dan pihak kedua merupakan pihak yang menyediakan layanan pengacakan.

Akuntabilitas MixCoin dibuat sebagai bukti transaksi. Apabila penyedia layanan berbuat curang dengan mencuri bitcoin milik pengguna, maka pengguna akan mengekspos tanda bukti transaksi tersebut, dengan demikian akan menghancurkan reputasi si penyedia layanan. Protokol MixCoin dapat dijelaskan dalam Gambar 19.



Gambar 23. Prosedur MixCoin [71]

Terdapat beberapa langkah yang harus dilakukan oleh pengguna A dan penyedia layanan M. Pengguna A membuat permintaan layanan kepada M untuk membuat transaksi bitcoin. Jika M setuju, maka M menandatangani informasi atas transaksi yang diminta oleh A dengan menggunakan private key milik M. Data yang telah ditandatangani ini merupakan bukti yang akan disimpan oleh A yang dapat diverifikasi oleh siapapun dengan menggunakan public key milik M. Berikutnya, A membayar sejumlah bitcoin yang telah disetujui kepada M, termasuk biaya transaksi yang dibayarkan kepada M. Apabila M bertindak jujur dengan mengirimkan bitcoin sebagaimana telah disetujui, maka tanda bukti dapat dihapus, namun apabila M bertindak curang, maka A dapat mempublikasikan bukti yang menyatakan bahwa M telah berbuat tidak jujur.

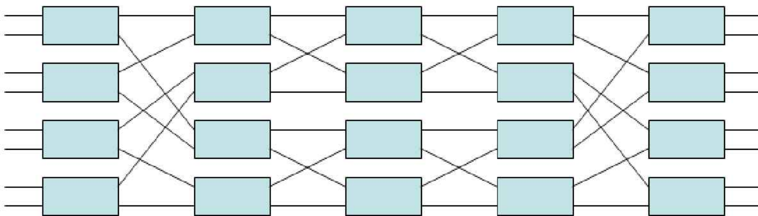
Merge Avoidance

Merge avoidance merupakan istilah yang dicetuskan oleh Mike Hearn dalam konsep yang diajukannya dalam mengidentifikasi masalah privasi bitcoin. Merge avoidance merupakan ide untuk memecah transaksi bitcoin dengan jumlah tertentu ke dalam beberapa transaksi untuk menghindari identifikasi pengguna atas jumlah spesifik yang dikirim dari satu pengguna ke pengguna lain [72]. Dengan pemecahan transaksi dengan jumlah bitcoin yang kecil, maka identifikasi transaksi tersebut akan lebih sulit dilakukan.

Merge avoidance ini bermanfaat pada kasus-kasus tertentu, misalnya Alice dan Bob bekerja dalam perusahaan yang sama dan digaji dalam bentuk bitcoin oleh perusahaan tersebut. Bob curiga bahwa ia mendapat gaji yang lebih kecil daripada Alice, oleh karena itu ia bisa saja meminta pada Alice untuk mengirimkan sejumlah kecil bitcoin. Dari situ Bob bisa menganalisis kira-kira transaksi mana yang jadi pembayaran gaji kepada Alice dan membuktikan kecurigaan Bob.

Circuit of Transactions

Sirkuit transaksi yang ditawarkan oleh Olivier Coutu dari Universitas Montreal Kanada dengan memanfaatkan teori jaringan untuk menyusun transaksi Bitcoin yang membentuk sirkuit kompleks, sehingga lebih sulit dilacak siapa identitas pihak-pihak yang terkait [73]. Terdapat beberapa konsep jaringan yang digunakan dalam sirkuti transaksi, di antaranya Benes Network seperti tampak pada Gambar 20.



Gambar 24. Benes Network yang Digunakan Pada Circuit of Transactions [74]

Sirkuit transaksi akan mempersulit analisis identitas atas para pemilik alamat Bitcoin yang terlibat dalam transaksi-transaksi tersebut, sehingga meningkatkan anonimitas Bitcoin.

Analisis Anonimitas Bitcoin

Terdapat beberapa karakteristik Bitcoin yang dijadikan dasar analisis transaksi Bitcoin maupun analisis identitas pemilik alamat Bitcoin. Para analis tentu saja dapat menggali informasi melalui blockchain Bitcoin yang terbuka dan dapat diakses oleh siapa saja. Lebih dari itu, informasi tambahan dapat digali oleh para analis untuk mempermudah pekerjaan mereka.

Alamat Bitcoin

Menyebarluaskan informasi alamat Bitcoin tentu saja menjadi hal yang lumrah agar dapat menerima pembayaran maupun menerima donasi. Namun para analis dapat secara langsung menghubungkan alamat Bitcoin tersebut dengan identitas asli si pemilik alamat. Transaksi yang terkait dengan alamat ini juga memiliki risiko privasi, tidak hanya pemilik alamat namun juga mereka yang bertransaksi dengan alamat tersebut.

Karakteristik informasi alamat Bitcoin yang dapat dikumpulkan di Internet dianalisis dalam sebuah riset [75]. Para periset mengumpulkan informasi alamat Bitcoin dari toko-toko yang mempublikasikan alamat Bitcoin milik mereka. Setelah itu, mereka mengelompokkan alamat-alamat tersebut berdasarkan lokasi geografis toko-toko tersebut. Tentu saja hal ini dapat disimpulkan bahwa pelanggan yang bertransaksi dengan alamat-alamat Bitcoin tersebut berada dalam lokasi geografis yang sama dengan toko-toko tersebut.

Untuk mengurangi kemungkinan analisis seperti di atas, para peneliti menyarankan agar alamat baru disediakan untuk setiap transaksi dengan pelanggan baru, sehingga dapat melindungi privasi pelanggan dengan sedikit lebih baik.

Pengguna Bitcoin yang memajang alamat Bitcoin mereka dalam situs-situs, forum, dan media sosial juga menjadi target analisis, termasuk juga alamat donasi Wikileaks [76]. Riset tersebut berhasil membuka relasi transaksi-transaksi bitcoin dalam grafik jaringan.

Input Jamak

Sebuah wallet Bitcoin dapat memiliki lebih dari 1 alamat Bitcoin yang mengelola alamat-alamat tersebut bagi pengguna. Jika pengguna mencoba untuk membuat transaksi yang membutuhkan dana dalam jumlah yang melebihi dana yang ada dalam 1 alamat, maka

wallet tersebut akan secara otomatis membuat transaksi dengan jumlah input yang lebih dari 1 (jamak). Konsep inilah yang digunakan oleh para analis dengan teknik klustering alamat dan mengasosiasikannya ke dalam identitas pemilik yang sama.

Karakteristik ini dieksploitasi dalam sebuah riset dengan melakukan teknik klustering [77]. Teknik ini dilakukan dengan mengumpulkan transaksi-transaksi dengan input jamak dan mengidentifikasikan pemilik alamat-alamat tersebut. Riset lain dengan jangkauan yang lebih luas juga dilakukan dengan menggunakan analisis kuantitatif untuk menentukan karakteristik umum transaksi-transaksi bitcoin yang telah terjadi [18].

Alamat Kembalian

Alamat kembalian (change address) merupakan sebuah alamat Bitcoin yang dimiliki oleh pengirim bitcoin untuk menerima selisih antara uang yang dimiliki dalam sebuah alamat dengan uang yang dibayarkan. Hal ini merupakan praktik yang umumnya dilakukan dalam transaksi Bitcoin. Dengan praktik ini, maka dapat dikatakan bahwa alamat kembalian dimiliki oleh pengirim bitcoin dalam sebuah transaksi.

Alamat kembalian juga digunakan dalam teknik klustering [77] dan diidentifikasi sebagai alamat yang dimiliki oleh pemilik yang sama dengan alamat input jamak.

Jumlah Bitcoin Yang Ditransaksikan

Jumlah bitcoin yang ada dalam transaksi Bitcoin dapat menjadi titik awal bagi analisis bitcoin. Hal inilah yang coba dihindari dalam teknik merge avoidance.

BAB 4 BITCOIN SCRIPTING

Dalam bab ini akan dibahas tentang cara membuat raw transaction (skrip transaksi) Bitcoin tanpa menggunakan wallet, namun menggunakan tool yang mempermudah proses pembuatan skrip transaksi.

BX

BX, singkatan dari Bitcoin Explorer, merupakan software toolkit yang dikembangkan oleh Eric Voskuil [78]. BX sendiri merupakan kelanjutan dari SX yang pertama kali dikembangkan oleh Pablo Martin dan Amir Taaki [69]. Kedua tool ini menggunakan perangkat yang sama, yakni libbitcoin, yakni library berbasis bahasa C++ yang dapat digunakan oleh pengembang untuk mempermudah usaha mereka dalam mengembangkan perangkat lunak pendukung bitcoin. Dengan menggunakan library Bitcoin, maka pengembang tidak perlu lagi mengimplementasikan operasi-operasi dasar Bitcoin, sehingga dapat lebih berfokus pada fungsionalitas software yang mereka kembangkan.

Berbeda dengan SX yang hanya mendukung sistem operasi Linux Ubuntu, BX memiliki beberapa versi sistem operasi seperti Mac OSX, Linux, dan Windows, sehingga mempermudah pengguna yang tidak terbiasa menggunakan Linux untuk belajar dasar-dasar Bitcoin. Tool BX dapat diunduh melalui alamat https://github.com/libbitcoin/libbitcoin-explorer/wiki/Download-BX. BX bersifat open source, sehingga kode sumber dapat dilihat dan digunakan secara gratis.

BX merupakan toolkit yang berbasis CLI (command line interface), namun demikian tidak sulit dipelajari. Tersedia juga cara pemakaian dan detail lebih lanjut untuk setiap perintah yang dapat dilihat dalam dokumen Github (https://github.com/libbitcoin/libbitcoin-explorer/wiki). BX versi Windows tidak memerlukan instalasi untuk digunakan. Cukup dengan mengunduh aplikasi, kemudian diakses melalui Command Prompt. Sebagai contoh, sebuah folder baru dapat dibuat di C:\BX yang berisi sebuah file dengan nama bx.exe.

Aplikasi BX yang digunakan dalam buku ini merupakan versi 2.1.0. Pembaca mungkin menggunakan versi yang lebih baru, tetapi perintah-perintah yang digunakan tidak akan jauh berbeda dengan yang dijelaskan di dalam buku.

Untuk mempermudah dalam konstruksi perintah, maka nama file **bx-windows-x64-icu-mainnet.exe** dapat diubah menjadi **bx.exe**, sehingga nanti format perintah berbentuk “*bx [perintah]*”. Format ini akan digunakan dalam penjelasan-penjelasan berikutnya.

Setting

BX memiliki setting standar bawaan aplikasi yang dapat ditampilkan melalui perintah "bx settings". Namun setting standar dapat diubah dengan membuat sebuah file bx.cfg dan diletakkan di dalam folder %ProgramData%\libbitcoin\bx.cfg. Berikut adalah contoh file konfigurasi BX.

```
# Bitcoin Explorer (BX) configuration file.

[general]

# Only hd-new and stealth-encode currently use the testnet distinction, apart from swapping servers.
# The network to use, either 'mainnet' or 'testnet'. Defaults to match the build.
network = mainnet
# The number of times to retry contacting a server or node, defaults to 0.
#connect_retries = 0
# The time limit for connection establishment, defaults to 5.
#connect_timeout_seconds = 5
# The time limit to complete the connection handshake, defaults to 30.
#channel_handshake_seconds = 30
# The path to the p2p hosts file, defaults to 'hosts.cache'.
#hosts_file = hosts.cache

[logging]
# The debug log file path, defaults to 'debug.log'.
#debug_file = debug.log
# The error log file path, defaults to 'error.log'.

#error_file = error.log

[mainnet]
# The URL of the mainnet Libbitcoin/Obelisk server.
#url = tcp://obelisk.airbitz.co:9091
url tcp://libbitcoin1.openbazaar.org:9091
# The Z85-encoded public key of the server certificate.
# server_cert_key =
# The path to the ZPL-encoded client private certificate file.
# cert_file =

[testnet]
# The URL of the Libbitcoin/Obelisk testnet server.
url = tcp://obelisk-testnet.airbitz.co:9091
# The Z85-encoded public key of the server certificate.
# server_cert_key =
# The path to the ZPL-encoded client private certificate file.
# cert_file =
```

Terdapat beberapa setting yang penting, di antaranya konfigurasi jaringan.

```
network = mainnet
```

Konfigurasi jaringan di atas digunakan untuk Mainnet yang tentu saja dapat diubah menjadi Testnet jika diperlukan.

```
network = testnet
```

Karena BX bergantung pada Libbitcoin/Obelisk server, maka jika server default tidak berjalan, maka perlu dilakukan modifikasi. Sebagai contoh, saat buku ini ditulis, server obelisk standar yang melayani mainnet sedang mengalami kerusakan, oleh sebab itu dilakukan modifikasi tcp://obelisk.airbitz.co:9091 menjadi tcp://libbitcoin1.openbazaar.org:9091.


```
#url = tcp://obelisk.airbitz.co:9091
url = tcp://libbitcoin1.openbazaar.org:9091
```

Tanda pagar (#) di depan parameter menunjukkan bahwa parameter tersebut tidak aktif.

Selain dengan memodifikasi konfigurasi standar, BX juga dapat menggunakan konfigurasi temporer dengan menambahkan parameter -c pada baris perintah. Sebagai contoh, dapat dibuat file konfigurasi testnet.cfg untuk mengeksplorasi jaringan testnet. File konfigurasi ini dapat diletakkan dalam folder yang sama dengan bx.exe.

```
C:\BX>bx fetch-height -c testnet.cfg
645774
```

Format Data Output

Terdapat beberapa format data yang dapat dipakai sebagai output sebuah perintah, yaitu info sebagai format standar, json, dan xml. Setting format data ini dapat dilakukan dengan parameter -f dalam perintah yang diinput. Berikut adalah contoh output dalam format json.

```
C:\BX>bx fetch-balance 141QU7S4tDMXQJK6ZvMvcuZDbk-
aBse4Se9 -f json
{
    "balance": {
        "address": "141QU7S4tDMXQJK6ZvMvcuZDbkaB-
se4Se9",
        "confirmed": "1290000",
        "received": "2240000",
        "unspent": "1290000"
    }
}
```

Sementara itu, berikut contoh output dalam format xml.

```
C:\BX>bx fetch-balance 141QU7S4tDMXQJK6ZvMvcuZDbk-
aBse4Se9 -f xml
<?xml version="1.0" encoding="utf-8"?>
<balance><address>141QU7S4tDMXQJK6ZvMvcuZDbkaB-
se4Se9</address><confirmed>1290000</confirmed><re-
ceived>2240000</received><unspent>1290000</un-
spent></balance>
```

Dengan format standar (tanpa parameter maupun dengan parameter -f info), contoh outputnya adalah sebagai berikut.

```
C:\BX>bx fetch-balance 141QU7S4tDMXQJK6ZvMvcuZDbk-
aBse4Se9
balance
{
    address 141QU7S4tDMXQJK6ZvMvcuZDbkaBse4Se9
    confirmed 1290000
    received 2240000
    unspent 1290000
}
```

Daftar Perintah dan Bantuan

Untuk mengetahui perintah apa saja yang didukung oleh BX, maka perintah "bx" dapat ditulis dalam command prompt. BX akan langsung menampilkan hasilnya tepat di bawah perintah yang dimasukkan oleh pengguna.

```
C:\BX>bx

Usage: bx COMMAND [--help]

Version: 2.1.0 [mainnet]

Info: The bx commands are:

address-decode
address-embed
address-encode
address-validate
base16-decode
base16-encode
base58-decode
base58-encode
base58check-decode
base58check-encode
base64-decode
base64-encode
bitcoin160
bitcoin256
btc-to-satoshi
cert-new
cert-public
ec-add
ec-add-secrets
ec-lock
ec-multiply
ec-multiply-secrets
ec-new
ec-to-address
```

```
ec-to-public
ec-to-wif
ec-unlock
fetch-balance
fetch-header
fetch-height
fetch-history
fetch-public-key
fetch-stealth
fetch-tx
fetch-tx-index
fetch-utxo
hd-new
hd-private
hd-public
hd-to-address
hd-to-ec
hd-to-public
hd-to-wif
help
input-set
input-sign
input-validate
message-sign
message-validate
mnemonic-new
mnemonic-to-seed
qrcode
ripemd160
satoshi-to-btc
script-decode
script-encode
script-to-address
seed
send-tx
send-tx-node
send-tx-p2p
settings
```

```
sha160
sha256
sha512
stealth-decode
stealth-encode
stealth-public
stealth-secret
stealth-shared
tx-decode
tx-encode
tx-sign
uri-decode
uri-encode
validate-tx
watch-address
watch-tx
wif-to-ec
wif-to-public
wrap-decode
wrap-encode

Bitcoin Explorer home page:

https://github.com/libbitcoin/libbitcoin-explorer
```

BX juga mendukung pipeline seperti dalam Linux. Pipe merupakan metode untuk memasukkan output dari sebuah perintah langsung ke perintah langsung. Dengan menggunakan pipe, pengguna dapat memasukkan beberapa perintah sekaligus jika output sebuah perintah digunakan sebagai input perintah lain.

Pipe menggunakan format "perintah_1 | perintah_2".

Selain itu terdapat juga perintah untuk menyimpan output sebuah perintah ke file eksternal dengan format "perintah > nama_file" atau dapat juga menggunakan file menjadi output sebuah perintah dengan format "perintah < nama_file".

Bitcoin OpCodes Dalam BX

Terdapat sedikit perbedaan dalam penyebutan OpCode dalam BX dengan yang telah dibahas dalam bab 2.4. Secara spesifik, script di dalam BX tidak memerlukan awalan OP_ ketika menunjuk sebuah OpCode tertentu. Sebagai contoh, OP_SHA256 akan disebut sebagai sha256 di dalam script BX. Lebih lanjut lagi, untuk menandai parameter, BX memanfaatkan tanda kurung persegi [dan]. Sebagai contoh, script

```
OP_SHA256 hash OP_EQUALVERIFY pubkey OP_CHECKSIG
```

akan ditulis sebagai

```
sha256 [ hash ] equalverify [ pubkey ] checksig
```

Perlu diperhatikan bahwa terdapat tanda spasi untuk memisahkan tanda kurung persegi dengan parameter yang ada di dalamnya. Jika tanda spasi tersebut tidak dibubuhkan, maka operasi dalam BX tidak akan berhasil.

Selain itu, di dalam pembuatan sebuah script, BX mengganti angka 0 dengan "zero" saat digunakan di dalam sebuah script. Untuk lebih mudahnya, contoh script berikut merupakan script di dalam fase redeem skema multisignature.

```
0 <signature 1> <signature 2> <P2SH script>
```

akan ditulis sebagai

zero <signature 1> <signature 2> <P2SH script>

Mengambil Informasi dari Blockchain

BX dilengkapi dengan fitur untuk mengambil beberapa informasi penting dari blockchain yang bermanfaat dalam pembuatan transaksi Bitcoin secara manual. Beberapa perintah akan dibahas berikut ini. Data yang diambil oleh BX diperoleh dari server Obelisk yang berbasis Libbitcoin.

Terkadang BX menghasilkan nilai timed out tergantung kondisi jaringan dan kondisi server yang melayani permintaan informasi. Oleh karena itu, apabila hal ini terjadi, maka perintah harus diulang sampai menampilkan hasil yang diminta.

Saldo Bitcoin

BX dapat mengambil informasi saldo yang terdapat dalam alamat Bitcoin tertentu dan menampilkannya dengan perintah "bx fetch-balance <alamat Bitcoin>".

```
  C:\BX>bx fetch-balance 141QU7S4tDMXQJK6ZvMvcuZDbk-
aBse4Se9
  balance
  {
      address 141QU7S4tDMXQJK6ZvMvcuZDbkaBse4Se9
      confirmed 1290000
      received 2240000
      unspent 1290000
  }
```


Informasi saldo sangat bermanfaat dalam pembuatan transaksi karena harus menentukan apakah sebuah alamat memiliki dana yang cukup untuk ditransaksikan. Informasi ini adalah informasi yang pertama kali harus diketahui oleh software wallet.

Histori Transaksi

Histori transaksi merupakan informasi transaksi yang telah lampau atas alamat Bitcoin tertentu. Informasi ini dapat diketahui dengan mengetikkan perintah "bx fetch-history <alamat Bitcoin>".

```
  C:\BX>bx  fetch-history  1JugYgC5PNF3JbrA1CUkozfWH-
6JE5Xy4YT
  transfers
  {
      transfer
      {
          received
          {
             hash f9fad10466c5de29c440ebebbdcf618a7b-
cc84f81efc55aa895b746387bc5f0e
              height 395524
              index 1
          }
```

```
    spent
          {
             hash e0e75b2c14f002defb928f1dd6953d83bd-
9baed1858f57fe283ac60ad2f5e424
              height 395918
              index 0
          }
          value 572500
      }
  }
```


Informasi di atas menunjukkan transaksi received (diterima) dan spent (dihabiskan). Transaksi received terjadi saat alamat tersebut menerima dana dari alamat lain, sementara transaksi spent terjadi saat alamat tersebut melakukan pembayaran ke alamat lain. Transaksi yang sudah terdapat informasi spent tidak dapat dipakai lagi.

Detail Transaksi

Informasi detail transaksi dapat diperoleh dengan menggunakan perintah "bx fetch-tx <transaction id>". Detail transaksi dapat digunakan untuk melihat informasi sebuah transaksi dengan lebih detail menggunakan transaction id. Informasi berupa input, output, ScriptSig, dan ScriptPubKey dapat dilihat dalam output perintah ini.

```
C:\BX>bx    fetch-tx    f9fad10466c5de29c440ebebbdc-
f618a7bcc84f81efc55aa895b746387bc5f0e
transaction
{
        hash  f9fad10466c5de29c440ebebbdcf618a7bc-
c84f81efc55aa895b746387bc5f0e
    inputs
    {
        input
        {
            address  1AmBUfow3RtXs19ZFrsDR7ZqWGY-
M8oUeN9
            previous_output
            {
        hash 247e028638d3302f129a8bc298bb067620f14f-
1b60aed32803f9bc4dc150e440
        index 1
    }
script      "[      304402203618fc8bd2dab5864399906836ce-
1ca417ff4560c52c7a9ff6ce52cbeadfadde022011219d850d-
25569e165f836488a0c8b125816f68ca9695eb7d5fa0b4419d5df801
]    [      02e7cd9010ba2923d910f87e2e500e88f1ceda72ea-
32da2e35291a098f8d43ea68 ]"
    sequence 4294967295
  }
}
lock_time 0
outputs
{
    output
    {
      address 1ESuXuXBQauyXBxhoh8ubarwUWd13ozMWP
          script "dup hash160 [ 937fe2ee82229d282edec-
2606c70e755875334c0 ] equalverify checksig"
      value 500000
    }
    output
    {
```

```
            address 1JugYgC5PNF3JbrA1CUkozfWH6JE5Xy4YT
             script "dup hash160 [ c470c7a017172e6efc-
fa2a008b63fb8f81e07919 ] equalverify checksig"
             value 572500
        }
    }
    version 1
}
```

Block Height

Block height merupakan nomor blok terbaru yang sudah ditemukan oleh miner. Untuk mengambil informasi block height terbaru, perintahnya adalah "bx fetch-height".

```
C:\BX>bx fetch-height
401619
```

Informasi block height dapat digunakan ketika membuat transaksi yang menggunakan fitur nLockTime atau CheckLockTimeVerify, di mana block height digunakan sebagai penanda waktu.

Operasi Hash

BX mendukung banyak tipe operasi hash yang dapat bermanfaat dalam pembuatan transaksi Bitcoin, namun hanya beberapa saja yang akan dibahas di sini.

SHA256

Untuk menghitung SHA256 atas sebuah nilai digunakan perintah "bx sha256 <nilai>". Sebagai contoh, nilai SHA256 dari "1234567890" adalah sebagai berikut.

```
C:\BX>bx sha256 1234567890
6c450e037e79b76f231a71a22ff40403f7d9b-
74b15e014e52
```

RIPEMD160

Nilai RIPEMD160 dapat dihitung menggunakan BX dengan perintah “bx ripemd160 <nilai>”. Sebagai contoh, nilai RIPEMD160 dari “1234567890” adalah sebagai berikut.

```
C:\BX>bx ripemd160 1234567890
06a57a74afecfa1500bb460b227c0a9ae4a3ba5c
```

BITCOIN160

Operasi bitcoin160 dalam BX merupakan gabungan 2 operasi hash, yakni RIPEMD160 atas hasil perhitungan SHA256 sebuah nilai. Untuk menghitung nilai bitcoin160 digunakan perintah “bx bitcoin160 <nilai>”. Sebagai contoh, nilai BITCOIN160 dari “1234567890” adalah sebagai berikut.

```
C:\BX>bx bitcoin160 1234567890
bcc38e43395ff5639e6db7d8736d59000bf8ab08
```


Selain cara di atas, bisa juga menggunakan perintah “bx sha256 [nilai] | bx ripemd160” yang akan menghasilkan nilai yang sama.

```
C:\BX>bx sha256 1234567890 | bx ripemd160
bcc38e43395ff5639e6db7d8736d59000bf8ab08
```

BITCOIN256

Operasi bitcoin256 merupakan gabungan beberapa operasi, yakni 2 kali operasi SHA256 ditambah dengan pengubahan urutan byte. Operasi bitcoin256 dapat dilakukan dengan menggunakan perintah “bx bitcoin256 <nilai>”. Sebagai contoh, nilai BITCOIN256 dari “1234567890” adalah sebagai berikut.

```
C:\BX>bx bitcoin256 1234567890
6936c0592eb03255b293ded434dd5d36bbd31b3e33e488b7e-
9468abba646e44e
```

Sebagai perbandingan, berikut adalah hasil dari operasi 2 kali SHA256

```
C:\BX>bx sha256 1234567890 | bx sha256
4ee446a6bb8a46e9b788a4333e1bd3bb365ddd34d4de93b-
25532b02e59c83669
```

Jika diperbandingkan di antara keduanya, maka terlihat bahwa nilai kedua merupakan little endian dari nilai pertama.

```
Nilai 1: 6936c0592eb03255b293ded434dd5d36bbd31b3e
33a488b7e9468abba646e44e
Nilai 2: 4ee446a6bb8a46e9b788a4333e1bd3bb365ddd34d4d
e9
```

Operasi bitcoin256 bermanfaat di antaranya untuk menghitung TxID (transaction ID) sebuah skrip transaksi Bitcoin.

Seed

Seed merupakan informasi acak yang dapat digunakan untuk membuat pasangan public key dan alamat Bitcoin baru. Sebuah seed dapat menghasilkan sebuah alamat, dan dengan demikian seed yang sama akan menciptakan alamat yang sama pula, oleh sebab itu sangat penting untuk menggunakan seed yang cukup baik.

Seed dapat dibuat oleh BX dengan sangat mudah menggunakan perintah "bx seed". BX sendiri dilengkapi oleh random generator, yakni algoritma untuk menciptakan data acak yang dapat membantu pengguna untuk membuat seed baru.

```
C:\BX>bx seed
1facd1162a165f09b303a497568ff1b3
```

Perintah di atas menghasilkan seed dengan nilai "1facd1162a165f09b303a497568ff1b3". Tentu saja untuk setiap perintah seed yang dimasukkan akan menghasilkan nilai seed yang berbeda-beda.

Membuat Alamat Bitcoin

Alamat Bitcoin dapat dibagi menjadi beberapa jenis, di antara alamat Mainnet, alamat Testnet, Hierarchical Deterministic address, dan Stealth address. Beberapa jenis alamat tersebut akan dibahas sebagai berikut.

Alamat Bitcoin Mainnet

Terdapat beberapa langkah untuk membuat alamat Bitcoin yang dapat digunakan dalam jaringan Mainnet yang melibatkan beberapa perintah, dengan langkah-langkah sebagai berikut.

- Buat seed baru dengan perintah "bx seed".

```
C:\BX>bx seed
a53e1f06ae07f38dd7df56d6eb4e773d
```

- Buat private key baru dari seed yang telah dibuat dengan perintah "bx ec-new <seed>".

```
C:\BX>bx ec-new a53e1f06ae07f38dd7df56d6eb4e773d
9297dab72fceb2c6d8d0198ccd0cecbeaa380950f25147a-
d46163a2d0cb7b9fa
```

- Buat public key dari private key yang telah dibuat dengan perintah "bx ec-to-public <private key>".

```
C:\BX>bx ec-to-public 9297dab72fceb2c6d8d0198ccd-
0cecbeaa380950f25147ad46163a2d0cb7b9fa
02bba6794bbd157fce30076a8746e96d5dc724e-
f0878a6b7282859e804404d6369
```

- Buat alamat Bitcoin dari public key yang telah dibuat dengan perintah "bx ec-to-address <public key>".

```
C:\BX>bx ec-to-address 02bba6794bbd157fce-
30076a8746e96d5dc724ef0878a6b7282859e804404d6369
1M7KB3MbsJgX3r8kEpwqMtR4qX7vZQvShT
```

Keempat langkah di atas dapat dipersingkat dengan menggunakan pipe. Tapi tentu saja private key harus tetap disimpan, sehingga membutuhkan 2 langkah saja.

- Buat private key baru

```
C:\BX>bx seed | bx ec-new
18498bedfe89530fe1ae82076254d2195980924e6f22653710
65b2975bf76c5e
```

- Buat alamat Bitcoin dari private key

```
C:\BX>bx ec-to-public 18498bedfe89530fe1ae82076254d2
195980924e6f2265371065b2975bf76c5e | bx ec-to-address
13bxGmdxFzfhKhQtTehfG9Bi5kHogCkxGJ
```

Hasil akhir alamat Bitcoin yang dibuat yakni `13bxGmdxFzfhKhQtTehfG9Bi5kHogCkxGJ`.

Alamat Bitcoin Testnet

Terdapat perbedaan alamat yang digunakan dalam Mainnet dan Testnet. Untuk membuat alamat Testnet dari key yang sama, maka digunakan parameter -v 111. Saat buku ini ditulis, Testnet yang ada merupakan Testnet3. Sementara untuk membuat alamat Mainnet tidak diperlukan parameter apapun, meskipun parameter -v 1 akan menghasilkan alamat Mainnet yang sama.

Dengan 2 langkah, alamat Bitcoin untuk Testnet dapat diciptakan.

- Buat private key baru.

```
C:\BX>bx seed | bx ec-new
c8b82c85eb9d0e0414931e54c73ff267134107247fb94ce-
12d7965e061852a84
```

- Buat alamat Bitcoin dari private key.

```
  C:\BX>bx        ec-to-public        c8b82c85eb-
9d0e0414931e54c73ff267
134107247fb94ce12d7965e061852a84 | bx ec-to-address -v
111
  n3oWSQosrZnPEsA8XUA8VwrSgdhcTcUfWW
```

Alamat Bitcoin Testnet memiliki awalan m atau n, sementara alamat Bitcoin Mainnet memiliki awalan 1.

Hierarchical Deterministic Address

Hierarchical Deterministic Address (HD Address) dapat dibuat dengan menggunakan BX. Yang dibutuhkan adalah sebuah seed sebagai titik awal. HD Address digunakan dalam HD Wallet yang telah dibahas dalam bab 2.3.3.

- Membuat seed baru.

Untuk membuat seed baru, dapat digunakan perintah "bx seed".


```
  C:\BX>bx seed
  1e25c85b9f5449cf9befed39bc2ab086
```

- Membuat parent HD private key.

Parent HD private key merupakan HD private key yang diturunkan langsung dari seed. Parent HD private key dapat digunakan untuk membuat banyak child HD private key.

```
  C:\BX>bx hd-new 1e25c85b9f5449cf9befed39bc2ab086
  xprv9s21ZrQH143K31TppoJeJeikip21Km7LatxxCmZ9jgT-
wYKghHSYsc5jEfaEpUWQ5k3VUEFKg6ESUHxaavKQdgA1G8TnqvZ-
ZADQ8W8erYYM7
```

- Membuat child HD private key.

Child HD private key dapat dibuat menggunakan sebuah parent HD private key. Sebuah child HD private key dapat digunakan sebagai parent HD private key untuk membuat child HD private key lainnya. Untuk membuat sebuah child HD private key, diperlukan parameter -i index, yakni sebuah angka integer dari 0 sampai dengan kira-kira

4 miliar. Jadi sebuah HD private key dapat menciptakan 4 miliar HD private key lainnya.

Untuk membuat sebuah child HD private key dari sebuah parent HD private key, digunakan perintah "bx hd-private –i <index> <private key>". Sebagai contoh, perintah berikut menggunakan index 100.000 dan parent HD private key yang dihasilkan dari langkah (b) di atas.

```
C:\BX>bx hd-private -i 100000 xprv9s21ZrQH143K31Tppo-
JeJeikip21Km7LatxxCmZ9jgTwYKghHSYsc5jEfaEpUWQ5k3VUE-
FKg6ESUHxaavKQdgA1G8TnqvZZADQ8W8erYYM7
xprv9uvCbePbZrk461NWGainMqKuDS9jkwtveiprEezRjW6X-
mZYoBjrZ7cNtWQy9JpPjCwQJTpA3eA94GD7HBKv8LEAa8zY6o-
gTs4qqjgwZttZh
```

- Membuat parent HD public key.

Informasi parent HD public key dapat dihitung dari parent HD private key. Sebagai contoh, menggunakan hasil operasi pada langkah (b) di atas pada perintah dengan format "bx hd-to-public <private key>".

```
C:\BX>bx hd-public xprv9s21ZrQH143K31TppoJeJeikip-
21Km7LatxxCmZ9jgTwYKghHSYsc5jEfaEpUWQ5k3VUEFKg6E-
SUHxaavKQdgA1G8TnqvZZADQ8W8erYYM7
xpub661MyMwAqRbcFVYHvpqefnfVGqrVjDqBx7tZ19x-
mJ1zvR81qpys89t3iWrj18E6w7Q5ecrabV9L2ho8QnW-
GJUhex5x7JJnUnTEwUg2Ts1Rr
```

- Membuat child HD public key.

Terdapat 2 cara untuk membuat child HD public key. Cara pertama adalah dengan menggunakan parent HD public key dengan perintah "bx hd-public <public key>" tambahan parameter –i index. Sebagai contoh akan digunakan parameter index 100.000 dan parent HD public key dari langkah (d).

```
C:\BX>bx hd-public -i 100000 xpub661MyMwAqRbcFVY-
HvpqefnfVGqrVjDqBx7tZ19xmJ1zvR81qpys89t3iWrj18E6w7Q5e-
crabV9L2ho8QnWGJUhex5x7JJnUnTEwUg2Ts1Rr
xpub68uZ19vVQEJMJVSyNcFniyGdmTzEAQcn1wkT33Q3H-
qdWeMswjHAofQhNMhxQgsUuHqfKEWXPnXcwz4BoQ7rgYyqx3W-
M4UK3EMCb6ifVWnmU
```

Cara kedua adalah dengan menggunakan child HD private key dengan perintah “bx hd-to-public <private key>”.

```
C:\BX>bx hd-to-public xprv9uvCbePbZrk461NWGain-
MqKuDS9jkwtveiprEezRjW6XmZYoBjrZ7cNtWQy9JpPjCwQJT-
pA3eA94GD7HBKv8LEAa8zY6ogTs4qqjgwZttZh
xpub68uZ19vVQEJMJVSyNcFniyGdmTzEAQcn1wkT33Q3H-
qdWeMswjHAofQhNMhxQgsUuHqfKEWXPnXcwz4BoQ7rgYyqx3W-
M4UK3EMCb6ifVWnmU
```


Kedua operasi di atas menghasilkan nilai yang sama.

- Mengubah HD key menjadi EC key.

Sayangnya, HD key baik itu public key maupun private key, tidak sama dengan EC key yang digunakan dalam operasi Bitcoin. Oleh sebab itu, setelah didapatkan HD key harus dikonversi menjadi EC key.

Untuk mengubah HD key menjadi EC key, baik itu private key maupun public key, dapat digunakan perintah “bx hd-to-ec <key>”. Sebagai contoh, HD private key dari langkah (b) dikonversi menjadi EC private key sebagai berikut

```
C:\BX>bx hd-to-ec xprv9s21ZrQH143K31TppoJeJeikip-
21Km7LatxxCmZ9jgTwYKghHSYsc5jEfaEpUWQ5k3VUEFKg6E-
SUHxaavKQdgA1G8TnqvZZADQ8W8erYYM7
0dcb9f0f1cf6067a8e23b7666e9fee4a2fe495eb6b58380b1d-
7f574a0fe24e02
```

Sementara, HD public key dari langkah (d) dapat dikonversi menjadi EC public key sebagai berikut.

```
C:\BX>bx hd-to-ec xpub661MyMwAqRbcFVYHvpqefnfVGqrV-
jDqBx7tZ19xmJ1zvR81qpys89t3iWrj18E6w7Q5ecrabV9L2ho-
8QnWGJUhex5x7JJnUnTEwUg2Ts1Rr
02cd79b364a4fd20dbb0369e4d7e155aea3001159bb908629b1f
78d555be997c89
```

- Membuat alamat Bitcoin.

Alamat Bitcoin dapat dibuat menggunakan HD private key maupun HD public key. Selain itu, setelah diketahui nilai EC public key, maka dapat juga digunakan untuk membuat alamat Bitcoin dengan menggunakan langkah-langkah seperti pada bab 4.9.1.

Untuk membuat alamat Bitcoin dengan menggunakan HD private key ataupun HD public key, maka digunakan perintah "bx hd-to-address <key>".

```
C:\BX>bx hd-to-address xprv9s21ZrQH143K31TppoJe-
Jeikip21Km7LatxxCmZ9jgTwYKghHSYsc5jEfaEpUWQ5k3VUEFK-
g6ESUHxaavKQdgA1G8TnqvZZADQ8W8erYYM7
1DdTHHDvj56XudxPZpgrQETGbtzP9sLit7
```

```
C:\BX>bx hd-to-address xpub661MyMwAqRbcFVYHvpqefn-
fVGqrVjDqBx7tZ19xmJ1zvR81qpys89t3iWrj18E6w7Q5ecrab-
V9L2ho8QnWGJUhex5x7JJnUnTEwUg2Ts1Rr
1DdTHHDvj56XudxPZpgrQETGbtzP9sLit7
```

Kedua perintah di atas menghasilkan alamat Bitcoin yang sama, sebab parameter key yang digunakan merupakan pasangan public-private key yang tepat.

Pay To Address Scripting

Transaksi Pay To Address (P2A) merupakan transaksi standar pembayaran kepada alamat Bitcoin tujuan. Pay To Address sering disebut juga dengan Pay To Public Key Hash (P2PKH). Pembuatan transaksi P2A akan dibahas dalam bagian ini.

Input dan Output Tunggal

Untuk menyederhanakan problem, pertama-tama akan dibahas pembuatan transaksi P2A yang memiliki 1 input dan 1 output. Langkah-langkahnya adalah sebagai berikut.

- Persiapan data

Beberapa data yang diperlukan dari pengirim adalah alamat Bitcoin, EC private key, dan EC public key, sementara data yang diperlukan dari penerima hanyalah alamat Bitcoin. Berikut adalah informasi yang digunakan dalam contoh, mengirim Bitcoin kembali ke alamat asal.

```
                              Pengirim
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa

                              Penerima
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
```

Contoh di atas hanya untuk kepentingan informatif belaka. Pada praktik sebenarnya, informasi private key tidak boleh dibagikan kepada orang lain. Pembaca sangat tidak disarankan menggunakan private key yang sama yang digunakan dalam buku ini dalam praktik pembelajaran.

ScriptPK (script public key atau scriptPubKey) akan digunakan dalam proses penandatanganan. Untuk transaksi Pay to Address, maka formatnya adalah

“`dup hash160 [ XYZ ] equalverify checksig`”

Nilai XYZ diperoleh dari operasi bitcoin160 atas public key, yakni dengan operasi “`bx bitcoin160 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534` “.

- Mengambil unspent transaction dan jumlah balance.

Sebagaimana telah dijelaskan, sebuah transaksi Bitcoin terdiri atas input dan output. Input transaksi Bitcoin merupakan rujukan atas transaksi sebelumnya yang belum pernah digunakan dalam transaksi Bitcoin manapun (unspent). Informasi tentang unspent transaction tersedia dalam histori transaksi.

```
C:\BX>bx fetch-history 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
transfers
{
    transfer
    {
        Received

        {
            hash
6211aaf8fe787f9a7ce8ce97a7752e68e765e62ad5a72279b0a5aa5bcb58a460
            height 401673
            index 0
        }
        value 20000
    }
}
```

Terdapat 2 informasi yang diperlukan, yakni hash value dan index. Hash value merupakan identitas transaksi (transaction ID atau TxID) yang menunjukkan transaksi tertentu di dalam blockchain Bitcoin. Sementara index merujuk pada urutan output dalam transaksi tersebut yang ditujukan ke alamat tertentu, yang dimulai dari angka 0. Value merupakan jumlah uang yang dikirimkan ke alamat tertentu dalam satuan satoshi. Dalam contoh di atas, informasi yang dapat diperoleh adalah sebagai berikut.

6211aaf8fe787f9a7ce8ce97a7752e68e765e62ad5a72279b0a5aa5bcb58a460	0	20000

Transaksi dengan TxID dan index di atas dapat diidentifikasi sebagai unspent transaction karena hanya memiliki informasi received tanpa informasi spent yang menyertainya.

- Menentukan biaya transaksi.

Biaya transaksi (transaction fee atau TxFee) tidak disebutkan secara eksplisit di dalam sebuah transaksi. Biaya transaksi merupakan selisih antara jumlah balance pada sisi input dan jumlah balance pada sisi output yang sederhananya dapat dituliskan sebagai berikut.

$$TxFee = \sum input - \sum output$$

Untuk transaksi yang hanya memiliki sebuah input dan sebuah output, maka 10.000 satoshi cukup sebagai biaya transaksi.

- Membuat unsigned transaction.

Unsigned transaction dapat dibuat menggunakan BX dengan perintah

“bx tx-encode –i <TXID>:<index> –o <alamat Bitcoin>:<jumlah>”

```
C:\BX>bx tx-encode -i 6211aaf8fe787f9a7ce8ce-97a7752e68e765e62ad5a72279b0a5aa5bcb58a460:0 -o 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa:10000
01000000160a458cb5baaa5b07922a7d52ae665e7682e-75a797cee87c9a7f78fef8aa1162000000000ffffff-f01102700000000000001976a9142d814b6ba2c3099aa5b3d38e-2cacfa13d28aaebe88ac00000000
```

Hasil dari operasi di atas yakni

```
01000000160a458cb5baaa5b07922a7d52ae665e7682e-75a797cee87c9a7f78fef8aa1162000000000ffffff-f01102700000000000001976a9142d814b6ba2c3099aa5b3d38e-2cacfa13d28aaebe88ac00000000
```


disebut sebagai unsigned transaction karena belum ada informasi tanda tangan dari pemilik uang yang akan membayarkan uangnya kepada pihak lain.

Contoh di atas menggunakan input dan output tunggal. Transaksi di atas dapat dibaca sebagai “transaksi sejumlah 10.000 satoshi dari pembayar kepada alamat Bitcoin 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa.

- Menandatangani unsigned transaction.

Penandatanganan transaksi Bitcoin memerlukan beberapa informasi, di antaranya private key, scriptPubKey, dan unsigned transaction yang telah dibuat. Perintah yang diperlukan berbentuk

“bx input-sign <private key> “<ScriptPubKey>” <unsigned transaction>”.

```
C:\BX>bx input-sign 23e9e2d5bc313418a9d5ca65ae-
ae181b0e77bfe400f79543494d703cb5a7be89 “dup
hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aae-
be ] equalverify checksig” 010000000160a458cb-
5baaa5b07922a7d52ae665e7682e75a797cee-
87c9a7f78fef8aa11620000000000fffffff-
f0110270000000000001976a9142d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe88ac00000000
3044022032f7a9af976a88bb1a-
b770eb1128031bf35298f1bdd4f86a287650de11a5cc0f-
02206051378d50e26e24332c9573296b784fb824b80358b795b-
9f3a6e4127ea6fedf01
```

Digital signature yang dihasilkan yakni

3044022032f7a9af976a88bb1a-b770eb1128031bf35298f1bdd4f86a287650de11a5cc0f-02206051378d50e26e24332c9573296b784fb824b80358b795b-9f3a6e4127ea6fedf01

Merupakan digital signature yang hanya dapat digunakan oleh transaksi terkait, dan tidak akan dapat digunakan dalam transaksi lainnya.

- Menambahkan digital signature ke dalam unsigned transaction.

Untuk menambahkan digital signature ke dalam unsigned transaction, maka perintah yang diperlukan adalah

```
“bx input-set “[ <signature> ] [ <public key> ]” <unsigned transaction>”
```

```
C:\BX>bx input-set "[ 3044022032f7a9af976a88b-
b1ab770eb1128031bf35298f1bdd4f86a287650de-
11a5cc0f02206051378d50e26e24332c9573296b-
784fb824b80358b795b9f3a6e4127ea6fedf01 ] [
0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534 ]" 01000000160a458cb5baaa5b07922a-
7d52ae665e7682e75a797cee87c9a7f78fef8aa1162000000000ffffff-
f0110270000000000001976a9142d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe88ac00000000
```

```
01000000160a458cb5baaa5b07922a7d52ae665e7682e-
75a797cee87c9a7f78fef8aa1162000000006a-
473044022032f7a9af976a88bb1ab770eb1128031bf
35298f1bdd4f86a287650de11a5cc0f-
02206051378d50e26e24332c9573296b784fb824b80358b795b-
9f3a6e4127ea6fedf01210372a523947fb9a160669b4b8b-
f905ab29d1ab906aa27901a2cd44aee17d7b8534ffffff-
f0110270000000000001976a9142d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe88ac00000000
```


Operasi di atas menghasilkan signed transaction sebagai berikut.

01000000160a458cb5baaa5b07922a7d52ae665e7682e-75a797cee87c9a7f78fef8aa1162000000006a-473044022032f7a9af976a88bb1ab770eb1128031bf35298f1bdd4f86a287650de11a5cc0f02206051378d50e26e24332c9573296b-784fb824b80358b795b9f3a6e4127ea6fedf01210372a523947f-b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534ffff-ffff0110270000000000001976a9142d814b6ba2c3099aa5b3d38e-2cacfa13d28aaebe88ac00000000

- Memvalidasi transaction.

Validasi transaksi digunakan untuk mengecek apakah semua input, output, dan digital signature telah disusun dengan benar. Perintahnya adalah "bx validate-tx <transaksi>".

```
C:\BX>bx validate-tx 01000000160a458cb5baaa5b07922a-
7d52ae665e7682e75a797cee87c9a7f78fef8aa1162000000006a-
473044022032f7a9af976a88bb1ab770eb1128031bf35298f1bdd4
f86a287650de11a5cc0f02206051378d50e26e24332c9573296b-
784fb824b80358b795b9f3a6e4127ea6fedf01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b853-
4ffffffff01102700000000000001976a9142d814b6ba-
2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
The transaction is valid.
```

Apabila perintah tersebut menghasilkan output "The transaction is valid", maka transaksi berhasil diverifikasi oleh aplikasi.

- Mengirim transaction ke jaringan.

Setelah transaksi selesai dibuat, maka tinggal mengirimnya ke jaringan. Terdapat 2 metode pengiriman transaksi, yakni send-tx dan send-tx-p2p. Perintah send-tx digunakan untuk mengirim transaksi melalui server Obelisk, sedangkan send-tx-p2p digunakan untuk mengirim transaksi melalui jaringan peer-to-peer.

```
C:\BX>bx   send-tx   01000000160a458cb5baaa5b07922a-
7d52ae665e7682e75a797cee87c9a7f78fef8aa1162000000006a-
473044022032f7a9af976a88bb1ab770eb1128031bf35298f1bdd4
f86a287650de11a5cc0f02206051378d50e26e24332c9573296b-
784fb824b80358b795b9f3a6e4127ea6fedf01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b853-
4ffffffff01102700000000000001976a9142d814b6ba-
2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
Sent transaction at 2016-Mar-11 09:02:15.
```

Untuk mengirim transaksi ke beberapa server, maka dapat digunakan parameter –n <jumlah server>, misalnya –n 10 akan mengirim transaksi ke 10 server yang terhubung dalam jaringan peer-to-peer.

```
C:\BX>bx send-tx-p2p -n 10 010000000160a458cb5baaa5b07922a-
7d52ae665e7682e75a797cee87c9a7f78fef8aa1162000000006a-
473044022032f7a9af976a88bb1ab770eb1128031bf35298f1bdd4f
86a287650de11a5cc0f02206051378d50e26e24332c9573296b-
784fb824b80358b795b9f3a6e4127ea6fedf01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b853-
4ffffffff0110270000000000001976a9142d814b6ba2c3099aa5b-
3d38e2cacfa13d28aaebe88ac00000000
Sent transaction at 2016-Mar-11 09:03:08.
Sent transaction at 2016-Mar-11 09:03:09.
Sent transaction at 2016-Mar-11 09:03:09.
Sent transaction at 2016-Mar-11 09:03:11.
Sent transaction at 2016-Mar-11 09:03:12.
Sent transaction at 2016-Mar-11 09:03:12.
Sent transaction at 2016-Mar-11 09:03:13.
Sent transaction at 2016-Mar-11 09:03:14.
Sent transaction at 2016-Mar-11 09:03:16.
Sent transaction at 2016-Mar-11 09:03:18.
```


Pengiriman transaction ke jaringan sebenarnya hanya perlu dilakukan 1 kali saja menggunakan 1 di antara 2 cara di atas. Namun demikian, terkadang terdapat permasalahan jaringan yang menyebabkan kedua cara tersebut harus dilakukan untuk memastikan transaksi terkirim.

- Menghitung nilai TxID.

TxID dapat dihitung meskipun raw transaction belum dikirim ke jaringan Bitcoin dengan menggunakan perintah “bx bitcoin256 <raw transaction>”.

```
C:\BX>bx bitcoin256 01000000160a458cb5baaa5b07922a-
7d52ae665e7682e75a797cee87c9a7f78fef8aa1162000000006a-
473044022032f7a9af976a88bb1ab770eb1128031bf35298f1bdd4
f86a287650de11a5cc0f02206051378d50e26e24332c9573296b-
784fb824b80358b795b9f3a6e4127ea6fedf01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b853-
4ffffffff01102700000000000001976a9142d814b6ba-
2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
8a25484723554a9c82e47b8e7641e10700dfda3226bbbbb-
7fbb5aee20decfc98
```

Hasilnya berupa sederet karakter yang merupakan TxID atas transaksi tersebut.

8a25484723554a9c82e47b8e7641e10700dfda3226bbbbb7fb-
b5aee20decfc98

- Mengecek konfirmasi transaksi.

Untuk mengecek apakah sebuah transaksi Bitcoin telah dikonfirmasi ke dalam sebuah blok, dapat digunakan perintah "bx fetch-tx-index <TxID>".

```
C:\BX>bx fetch-tx-index 8a25484723554a9c82e-
47b8e7641e10700dfda3226bbbbb7fbb5aee20decfc98
metadata
{
        hash 8a25484723554a9c82e47b8e7641e10700df-
da3226bbbbb7fbb5aee20decfc98
     height 402084
     index 473
}
```

Berdasarkan informasi di atas, transaksi dengan TxID berikut:

8a25484723554a9c82e47b8e7641e10700dfda3226bbbbb7fb-
b5aee20decfc98

telah dikonfirmasi ke dalam blok dengan nomor 402084 urutan nomor 473 dalam blok tersebut.

Input dan Output Jamak

Transaksi bitcoin dengan menggunakan input datn output jamak merupakan transaksi yang memiliki lebih dari 1 TxIn dan lebih dari 1 TxOut. Sebagai contoh misalnya alamat asal 159cJBQeWyLx9hP6hz-3tYBc3mvHAzn8Goa memiliki bitcoin sejumlah 90.000 satoshi yang berasal dari 2 transaksi yang lalu, ingin mengirim bitcoin sejumlah 75.000 satoshi kepada 1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr dan sisa bitcoin sebesar 5.000 satoshi akan dikembalikan ke alamat 1CQNgBtYwD8qHthBvSXrbtrFnVcX1TBXrb yang merupakan milik pengirim.

- Persiapan data.

Informasi yang perlu disiapkan sebagai berikut.

```
                                   Pengirim
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa

                                   Penerima
Private key: c8daee2ac304d64d9fe77ee30b7b092e611ea4d314d54727e14c3ba4dbf4ec24
Public key : 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77
ScriptPK    : dup hash160 [af4775b9355b39a7b2798bcdba1618ae89b679c6 ] equalverify checksig
Address    : 1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr

                                   Kembalian
Private key: bc9bec66f7eaff9f383091aedde72b27667cc27760fc0cce09c8f6932f5fbc86
Public key : 03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9
ScriptPK    : dup hash160 [7d150b6f24d21cc85114cee007b745b473e877c0 ] equalverify checksig
Address    : 1CQNgBtYwD8qHthBvSXrbtrFnVcX1TBXrb
```

- Mengambil unspent transaction dan jumlah balance.

Unspent transaction dapat dilihat menggunakan perintah "bx fetch-history".

```
  C:\BX>bx  fetch-history  159cJBQeWyLx9hP6hz3tYBc3m-
vHAzn8Goa
  transfers
  {
      transfer
      {
          received
          {
                    hash  f4598b6011fd741903cd7ee9ebe-
778f845e6a740287915d38d81162059ef4630
              height 402087
              index 1
```

```
      }
          value 80000
      }
      transfer
      {
          received
          {
           hash 8a25484723554a9c82e47b8e7641e10700df-
da3226bbbbb7fbb5aee20decfc98
              height 402084
              index 0
          }
          value 10000
      }
      transfer
      {
          received
          {
          hash 6211aaf8fe787f9a7ce8ce97a7752e68e765e-
62ad5a72279b0a5aa5bcb58a460
              height 401673
              index 0
          }
          spent
          {
           hash 8a25484723554a9c82e47b8e7641e10700df-
da3226bbbbb7fbb5aee20decfc98
              height 402084
              index 0
          }
          value 20000
      }
  }
```

Terdapat 2 unspent transaction, yakni:

f4598b6011fd741903cd7ee9ebe778f845e6a740287915d38d81162059ef4630	1	80000
8a25484723554a9c82e47b8e7641e10700dfda3226bbbbb7fbb5aee20decfc98	0	10000

Nilai total balance dapat diperoleh dengan perintah "bx fetch-balance"

```
C:\BX>bx fetch-balance 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
balance
{
    address 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
    confirmed 90000
    received 110000
    unspent 90000
}
```

Menurut informasi di atas, kini alamat 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa memiliki dana sebesar 90.000 satoshi.

- Menentukan biaya transaksi.

Untuk menyederhanakan perhitungan, biaya transaksi dapat ditentukan sebesar 10.000 satoshi.

- Membuat unsigned transaction.

Input dan output jamak dapat dibuat menggunakan BX dengan format sebagai berikut.

"bx tx-encode –i <TXID1>:<index1> –i <TXID2>:<index2> –i <TXID3>:<index3> –o <alamat1>:<jumlah1> –o <alamat2>:<jumlah2> –o <alamat3>:<jumlah3>"

```
 C:\BX\bx tx-encode -i f4598b6011fd741903c-
d7ee9ebe778f845e6a740287915d38d81162059ef4630:1 -i
8a25484723554a9c82e47b8e7641e10700dfda3226bbbbb7fb-
b5aee20decfc98:0 -o 1GynrgvHWJKJQwskZS2gRzMntEVneC-
j1sr:75000 -o 1CQNgBtYwD8qHthBvSXrbtrFnVcX1TBXrb:5000
 01000000023046ef592016818dd315792840a7e645f878e7e-
be97ecd031974fd11608b59f40100000000ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a0000000000ffffffff-
02f8240100000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac881300000000000001976a9147d150b6f24d-
21cc85114cee007b745b473e877c088ac00000000
```

Hasil operasi tersebut merupakan unsigned transaction, yaitu:

```
0100000023046ef592016818dd315792840a7e645f878e7e-
be97ecd031974fd11608b59f40100000000ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a0000000000ffffffff-
02f824010000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac8813000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
```

- Menandatangani unsigned transaction.

Karena terdapat 2 input, maka dibutuhkan 2 digital signature pula, yakni 1 untuk setiap input. Hal ini berlaku untuk setiap transaksi Bitcoin. Perintah yang dapat digunakan adalah

```
"bx input-sign –i <index> <private key> "<ScriptPub-
Key>" <unsigned transaction>".
```

Terdapat sedikit perbedaan dengan transaksi yang hanya memiliki 1 input saja, yakni adanya parameter -i yang merupakan parameter index, yaitu posisi input dalam unsigned transaction yang telah dibuat sebelumnya. Posisi input dimulai dari angka 0, bukan 1.

```
C:\BX>bx input-sign -i 0 23e9e2d5bc313418a9d-
5ca65aeae181b0e77bfe400f79543494d703c-
b5a7be89 "dup hash160 [ 2d814b6ba2c3099aa5b-
3d38e2cacfa13d28aaebe ] equalverify checksig"
0100000023046ef592016818dd315792840a7e645f878e7e-
be97ecd031974fd11608b59f40100000000ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a0000000000ffffffff-
02f824010000000001976a914af4775b-
9355b39a7b2798bcdba1618ae89b679c688a-
c8813000000000001976a9147d150b6f24d21cc85114cee007b-
745b473e877c088ac00000000
30440220475cab8a2cff015c6c9f3766c017df4e18fa-
c87e858355e43f54424584cd06ec02201fb83e26e2f-
538433c94f344945f1a79231aeb7866c5c1fdb1b6fb6c-
4679da4f01
```

Berikut adalah signature untuk index 0.

```
30440220475cab8a2cff015c6c9f3766c017df4e18fa-
c87e858355e43f54424584cd06ec02201fb83e26e2f538433c94f-
344945f1a79231aeb7866c5c1fdb1b6fb6c4679da4f01
```

Sementara untuk index 1

```
C:\BX>bx input-sign -i 1 23e9e2d5bc313418a9d-
5ca65aeae181b0e77bfe400f79543494d703c-
b5a7be89 "dup hash160 [ 2d814b6ba2c3099aa5b-
3d38e2cacfa13d28aaebe ] equalverify checksig"
0100000023046ef592016818dd315792840a7e645f878e7e-
be97ecd031974fd11608b59f40100000000ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a000000000ffffffff-
02f82401000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac8813000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
3044022074a8808a85cc3ff6e652a5dd95ba6a-
0727f5e60627ead2df302b0bd63a8e0a2702205396716125d-
04c4127363ed5ffa96bdfe699f13fe9953e1b0aabf-
1de824c104e01
```

Yang menghasilkan signature:

```
3044022074a8808a85cc3ff6e652a5dd95ba6a0727f5e60627e-
ad2df302b0bd63a8e0a2702205396716125d04c4127363ed5f-
fa96bdfe699f13fe9953e1b0aabf1de824c104e01
```

- Menambahkan digital signature ke dalam unsigned transaction.

Digital signature yang telah dibuat dapat digabungkan ke dalam unsigned transaction satu-persatu.

```
"bx input-set –i <index> "[ <signature> ] [ <public
key> ]" <unsigned transaction>"
```

Index tidak dimulai dari 1 ke 2 dan 3, melainkan dimulai dari 0, kemudian 1, dan 2.

```
C:\BX>bx input-set -i 0 "[ 30440220475cab8a2cff-
015c6c9f3766c017df4e18fac87e858355e43f-
54424584cd06ec02201fb83e26e2f538433c94f-
344945f1a79231aeb7866c5c1fdb1b6fb6c4679da4f01 ] [
0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534]"01000000023046ef592016818dd315792840a7e645f878e7e-
be97ecd031974fd11608b59f40100000000ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a000000000fffffff-
02f8240100000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac881300000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000

01000000023046ef592016818dd315792840a7e-
645f878e7ebe97ecd031974fd11608b59f4010000006a-
4730440220475cab8a2cff015c6c9f3766c017df4e18fa-
c87e858355e43f54424584cd06ec02201fb83e26e2f-
538433c94f344945f1a79231aeb7866c5c1fdb1b6fb6c-
4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a000000000fffffff-
02f8240100000000001976a914af4775b-
9355b39a7b2798bcdba1618ae89b679c688a-
c881300000000000001976a9147d150b6f24d21c-
c85114cee007b745b473e877c088ac00000000
```

Hasil operasi ini sebagai berikut:

```
01000000023046ef592016818dd315792840a7e645f878e7e-
be97ecd031974fd11608b59f4010000006a4730440220475ca-
b8a2cff015c6c9f3766c017df4e18fac87e858355e43f544245-
84cd06ec02201fb83e26e2f538433c94f344945f1a79231aeb-
7866c5c1fdb1b6fb6c4679da4f01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534ffffffff98fcec0de2aeb5fbb7bbbb2632dad-
f0007e141768e7be4829c4a55234748258a000000000fffffff-
02f8240100000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac881300000000000001976a9147d150b6f-
```

dimaz ankaa wijaya

```
24d21cc85114cee007b745b473e877c088ac00000000
```

Akan digunakan sebagai parameter <unsigned transaction> pada proses penambahan digital signature berikutnya, yakni untuk index 1.

```
  C:\BX>bx  input-set  -i  1  “[  3044022074a8808a85c-
c3ff6e652a5dd95ba6a0727f5e60627ead2df302b0b-
d63a8e0a2702205396716125d04c4127363ed5f-
fa96bdfe699f13fe9953e1b0aabf1de824c104e01      ]      [
0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534       ]”   0100000023046ef592016818d-
d315792840a7e645f878e7ebe97ecd031974fd11608b-
59f4010000006a47304402204 75cab8a2cff015c6c9f-
3766c017df4e18fac87e858355e43f54424584cd06ec02201f-
b83e26e2f538433c94f344945f1a79231aeb7866c5c1fdb1b-
6fb6c4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98f-
cec0de2aeb5fbb7bbbb2632dadf0007e141768e-
7be4829c4a55234748258a0000000000ffffffff-
02f82401000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac881300000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
  0100000023046ef592016818dd315792840a7e-
645f878e7ebe97ecd031974fd11608b59f4010000006a-
47304402204 75cab8a2cff015c6c9f3766c017df4e18fa-
c87e858355e43f54424584cd06ec02201fb83e26e2f-
538433c94f344945f1a79231aeb7866c5c1fdb1b6fb6c-
4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98fc-
ec0de2aeb5fbb7bbbb2632dadf0007e141768e7be4829c4a
55234748258a000000006a473044022074a8808a85cc3f-
f6e652a5dd95ba6a0727f5e60627ead2df302b0bd63a8e0a-
2702205396716125d04c4127363ed5ffa96bdfe699f13fe9953e-
1b0aabf1de824c104e01210372a523947fb9a160669b4b8b-
f905ab29d1ab906aa27901a2cd44aee17d7b8534ffffffff02-
f82401000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac881300000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
```

Hasil akhir berupa signed transaction.

```
0100000000023046ef592016818dd315792840a7e-
645f878e7ebe97ecd031974fd11608b59f4010000006a-
4730440220475cab8a2cff015c6c9f3766c017df4e18fa-
c87e858355e43f54424584cd06ec02201fb83e26e2f-
538433c94f344945f1a79231aeb7866c5c1fdb1b6fb6c-
4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98fc-
ec0de2aeb5fbb7bbbb2632dadf0007e141768e7be4829c4a
55234748258a000000006a47304402207 4a8808a85cc3f-
f6e652a5dd95ba6a0727f5e60627ead2df302b0bd63a8e0a-
2702205396716125d04c4127363ed5ffa96bdfe699f13fe9953e-
1b0aabf1de824c104e01210372a523947fb9a160669b4b8b-
f905ab29d1ab906aa27901a2cd44aee17d7b8534ffffffff02-
f824010000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac8813000000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
```

- Memvalidasi transaction.

```
C:\BX>bx    validate-tx    01000000023046ef592016818d-
d315792840a7e645f878e7ebe97ecd031974fd11608b-
59f4010000006a4730440220475cab8a2cff015c6c9f-
3766c017df4e18fac87e858355e43f54424584cd06ec02201f-
b83e26e2f538433c94f344945f1a79231aeb7866c5c1fdb1b-
6fb6c4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98fcec0de-
2aeb5fbb7bbbb2632dadf0007e141768e7be4829c4a552347482 5
8a000000006a47304402207 4a8808a85cc3ff6e652a5dd95ba6a-
0727f5e60627ead2df302b0bd63a8e0a2702205396716125d-
04c4127363ed5ffa96bdfe699f13fe9953e1b0aabf1de-
824c104e01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff-
02f824010000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac8813000000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
The transaction is valid.
```

- Mengirim transaction ke jaringan.

```
  C:\BX>bx     send-tx     0100000023046ef592016818d-
d315792840a7e645f878e7ebe97ecd031974fd11608b-
59f4010000006a4730440220475cab8a2cff015c6c9f-
3766c017df4e18fac87e858355e43f54424584cd06ec02201f-
b83e26e2f538433c94f344945f1a79231aeb7866c5c1fdb1b-
6fb6c4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98fcec0de-
2aeb5fbb7bbbb2632dadf0007e141768e7be4829c4a5523474825
8a000000006a473044022074a8808a85cc3ff6e652a5dd95ba6a-
0727f5e60627ead2df302b0bd63a8e0a2702205396716125d-
04c4127363ed5ffa96bdfe699f13fe9953e1b0aabf1de-
824c104e01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff-
02f8240100000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac88130000000000001976a9147d150b6f-
24d21cc85114cee007b745b473e877c088ac00000000
  Sent transaction at 2016-Mar-11 10:58:50.
```


- Menghitung nilai TxID.

```
  C:\BX>bx     bitcoin256     0100000023046ef592016818d-
d315792840a7e645f878e7ebe97ecd031974fd11608b-
59f4010000006a4730440220475cab8a2cff015c6c9f-
3766c017df4e18fac87e858355e43f54424584cd06ec02201f-
b83e26e2f538433c94f344945f1a79231aeb7866c5c1fdb1b-
6fb6c4679da4f01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff98fcec0de-
2aeb5fbb7bbbb2632dadf0007e141768e7be4829c4a5523474825
8a000000006a473044022074a8808a85cc3ff6e652a5dd95ba6a-
0727f5e60627ead2df302b0bd63a8e0a2702205396716125d-
04c4127363ed5ffa96bdfe699f13fe9953e1b0aabf1de-
824c104e01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffff-
02f8240100000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac88130000000000001976a9147d150b6f24d-
21cc85114cee007b745b473e877c088ac00000000
  91ba44d4f3c9a63371fdd47373671e3a08fe968bf-
0b49e16301928c8cd93f028
```

- Mengecek konfirmasi transaksi.

```
  C:\BX>bx    fetch-tx-index    91ba44d4f3c9a63371fd-
d47373671e3a08fe968bf0b49e16301928c8cd93f028
  metadata
  {
      hash 91ba44d4f3c9a63371fdd47373671e3a08fe968bf-
0b49e16301928c8cd93f028
      height 402097
      index 973
  }
```

Null Script Scripting

Seperti telah dijelaskan di dalam bagian 2.2.6, transaksi Null script tidak mentransaksikan bitcoin dari alamat yang satu ke alamat yang lain. Sebuah transaksi Bitcoin hanya dapat memiliki sebuah null script yang menjadi output transaksi tersebut. Meskipun null script termasuk salah satu jenis transaksi yang penting, namun sayangnya BX tidak memiliki perintah khusus untuk membuat null script, sehingga harus dibuat secara manual. Komponen utama dari transaksi null script menggunakan OP_RETURN dengan format OP_RETURN <pesan>.

Meskipun cukup sulit, belajar menyusun null script akan meningkatkan pengetahuan dalam Bitcoin scripting secara lebih detail. Proses pembuatan null script terbagi ke dalam beberapa tahap.

- Persiapan data

```
                                 Pengirim
Private key: c8daee2ac304d64d9fe77ee30b7b092e611ea4d314d54727e14c3ba4dbf4ec24
Public key : 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77
ScriptPK   : dup hash160 [ af4775b9355b39a7b2798bcdba1618ae89b679c6 ] equalverify checksig
Address    : 1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr

                                 Penerima
Address    : 1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr
```

Kemudian perlu diketahui unspent transaction yang dimiliki alamat tersebut dengan cara yang sama sebagaimana telah dibahas pada 4.10.1 dan 4.10.2. Unspent transaction sebagai berikut:

[redacted]		
91ba44d4f3c9a63371fdd47373671e3a08fe968bf0b49e16301928c8cd93f028	0	75000

- Persiapan pesan

Pesan yang dapat disisipkan menggunakan null script dapat berupa apa saja, namun tetap terdapat batasan panjang transaksi null script sebesar maksimum 80 byte. Sebuah karakter diwakili oleh 1 byte, oleh karena itu panjang pesan yang dapat disisipkan sebesar kurang lebih 80 byte, masih harus dikurangi beberapa byte untuk menyisipkan perintah. Untuk menghindari transaksi ditolak oleh jaringan Bitcoin, ada baiknya panjang pesan dibatasi sepanjang 75 karakter.

Pesan sebagai berikut akan dipakai dalam contoh kali ini. Kedua tanda petik tidak digunakan.

"Belajar menyusun transaksi Bitcoin secara manual dengan menggunakan BX. Dimaz"

Pesan di atas harus diubah ke dalam format BASE16 dengan perintah **"bx base16-encode <pesan>"**

```
C:\BX>bx base16-encode "Belajar menyusun transaksi Bitcoin secara manual dengan menggunakan BX. Dimaz"
42656c616a6172206d656e797573756e207472616e73616b736920426974636f696e20736563617261206d616e75616c2064656e67616e206d656e6767756e616b616e2042582e2044696d617a
```


Setelah itu, pesan berformat BASE16 tersebut akan dimasukkan ke dalam perintah OP_RETURN dengan format "bx script-encode "return [<pesan berformat base16>]" ".

```
C:\BX>bx script-encode "return [ 42656c616a6172206d656e797573756e207472616e73616b736920426974636f696e20736563617261206d616e75616c2064656e67616e206d656e6767756e616b616e2042582e2044696d617a ]"
6a4c4d42656c616a6172206d656e797573756e207472616e73616b736920426974636f696e20736563617261206d616e75616c2064656e67616e206d656e6767756e616b616e2042582e2044696d617a
```

Hasilnya yaitu:

6a4c4d42656c616a6172206d656e797573756e20747261

```
6e73616b736920426974636f696e2073656361726120616e-
75616c2064656e67616e206d656e6767756e616b616e2042582e20-
44696d617a
```

Setelah itu, perlu diketahui jumlah byte skrip di atas yang direpresentasikan dalam heksadesimal. Setiap byte direpresentasikan ke dalam 2 karakter script. Dengan panjang script di atas yaitu 160 karakter, maka jumlah byte script di atas adalah 80 byte. Untuk mengubah bilangan desimal menjadi heksadesimal dapat digunakan kalkulator PROGRAMMER yang tersedia dalam sistem operasi Windows.



Gambar 25. Kalkulator Programmer

Dari perhitungan di atas, didapatkan nilai 50 yang merupakan representasi heksadesimal dari angka desimal 80. Angka ini ditambahkan di depan skrip di atas, menghasilkan skrip sebagai berikut.

```
506a4c4d42656c616a6172206d656e797573756e20747261
6e73616b736920426974636f696e2073656361726120616e-
75616c2064656e67616e206d656e6767756e616b616e2042582e-
2044696d617a
```

- Membuat kerangka unsigned transaction.

Untuk mempermudah proses pembuatan transaksi null script, maka sebuah transaksi temporer akan dibuat.

```
C:\BX>bx tx-encode -i 91ba44d4f3c9a63371fd-d47373671e3a08fe968bf0b49e16301928c8cd93f028:0 -o 1Gyn-rgvHWJKJQwskZS2gRzMntEVneCj1sr:0 -o 1GynrgvHWJKJQwskZS-2gRzMntEVneCj1sr:65000
01000000000128f093cdc8281930169eb4f08b96fe-083a1e677373d4fd7133a6c9f3d444ba910000000000ffffff-f02000000000000000001976a914af4775b9355b39a7b2798bcd-ba1618ae89b679c688ace8fd000000000000001976a914af4775b-9355b39a7b2798bcdba1618ae89b679c688ac00000000
```

Transaksi temporer di atas seolah-olah mengirimkan bitcoin ke 2 alamat yang sama, yakni `1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr` dengan nilai 0 satoshi dan 65.000 satoshi , yang merupakan nilai input setelah dikurangi biaya transaksi sebesar 10.000 satoshi. Output pertama atas transaksi tersebut akan dimodifikasi agar dapat mencantumkan null script.

- Dekonstruksi transaksi

Transaksi yang dihasilkan dari langkah sebelumnya adalah sebagai berikut.

```
01000000000128f093cdc8281930169eb4f08b96fe-083a1e677373d4fd7133a6c9f3d444ba910000000000ffffff-f02000000000000000001976a914af4775b9355b39a7b2798bcd-ba1618ae89b679c688ace8fd000000000000001976a914af4775b-9355b39a7b2798bcdba1618ae89b679c688ac00000000
```

Transaksi tersebut dapat didekonstruksi seperti berikut.

No	Kode	Makna
1	01000000	Versi protocol Bitcoin (1) dalam format *little endian*
2	01	Jumlah input (1)
3	28f093cdc8281930169eb4f08b96fe08 3a1e677373d4fd7133a6c9f3d444ba91	TxID input dalam format *little endian*
4	00000000	Index TxID
5	00	Tempat tanda tangan
6	ffffffff	Sequence Number dalam format *little endian*
7	02	Jumlah output (2)
8	0000000000000000	Jumlah yang ditransfer ke output ke-1 dalam format *little endian*
9	19	Jumlah byte informasi output ke-1
10	76	OP_DUP
11	a9	OP_HASH160
12	14	Jumlah byte yang akan dimasukkan ke dalam stack
13	af4775b9355b39a7b2798bcdba1618ae 89b679c6	Nilai hash public key output ke-1
14	88	OP_EQUALVERIFY
15	ac	OP_CHECKSIG
16	e8fd000000000000	Jumlah yang ditransfer ke output ke-2 dalam format little endian
17	19	Jumlah byte informasi output ke-2
18	76	OP_DUP
19	a9	OP_HASH160
20	14	Jumlah byte yang akan dimasukkan ke dalam stack
21	af4775b9355b39a7b2798bcdba1618ae 89b679c6	Nilai hash public key output ke-2
22	88	OP_EQUALVERIFY
23	ac	OP_CHECKSIG
24	00000000	LockTime dalam format *little endian*

- Modifikasi transaksi

Baris ke-9 sampai dengan baris ke-15 dari langkah (d) di atas akan dimodifikasi dengan script OP_RETURN yang telah disiapkan pada langkah (b).

No	Kode	Makna
1	01000000	Versi protocol Bitcoin (1) dalam format *little endian*
2	01	Jumlah input (1)
3	28f093cdc8281930169eb4f08b96fe08 3a1e677373d4fd7133a6c9f3d444ba91	TxID input dalam format *little endian*.
4	00000000	Index TxID
5	00	Tempat tanda tangan
6	ffffffff	Sequence Number
7	02	Jumlah output (2)
8	0000000000000000	Jumlah yang ditransfer ke output ke-1
9-15	506a4c4d42656c616a6172206d656e79 7573756e207472616e73616b73692042 6974636f696e2073656361726120 6d61 6e75616c2064656e67616e206d656e67 67756e616b616e2042582e2044696d61 7a	Transaksi OP_RETURN
16	e8fd000000000000	Jumlah yang ditransfer ke output ke-2 dalam format *little endian*.
17	19	Jumlah byte informasi output ke-2
18	76	OP_DUP
19	a9	OP_HASH160
20	14	Jumlah byte yang akan dimasukkan ke dalam stack
21	af4775b9355b39a7b2798bcdba1618ae 89b679c6	Nilai hash public key output ke-2
22	88	OP_EQUALVERIFY
23	ac	OP_CHECKSIG
24	00000000	LockTime dalam format *little endian*

- Rekonstruksi transaksi

Setelah direkonstruksi dengan menggabungkan semua komponen secara berurutan, informasi dari langkah (e) di atas dapat diperoleh unsigned transaction sebagai berikut:

```
0100000001 28f093cdc8281930169eb-
4f08b96fe083a1e677373d4fd7133a6c9f3d-
444ba9100000000000ffffffff020000000000000000506a4c-
4d42656c616a6172206d656e797573756e207472616e73616b-
73692042697463 6f696e20736563617261206d616e-
75616c2064656e67616e206d656e6767756e616b616e2042582e-
2044696d617ae8fd0000000000001976a914af4775b9355b39a7b-
2798bcdba1618ae89b679c688ac00000000
```

- Menandatangani transaksi

```
C:\BX>bx input-sign c8daee2ac304d64d9fe77ee30b-7b092e611ea4d314d54727e14c3ba4dbf4ec24 “dup hash160 [ af4775b9355b39a7b2798bcdba1618ae89b679c6 ] equalverify checksig” 01000000128f093cd-c8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d-444ba910000000000ffffffff0200000000000000000506a4c-4d42656c616a6172206d656e797573756e207472616e73616b-7369204269746636f696e20736563617261206d-616e75616c2064656e67616e206d656e6767756e-616b616e2042582e2044696d617ae8fd000000000001976a91-4af4775b9355b39a7b2798bcdba1618ae89b-679c688ac00000000
30450221009ee39686b437f8d2f57e141192653d-84481ba86f2fba2dae0416d756f34477bd022039c-225979716397d37cd77c3ea6a1abd75fe43c2d736bcc7f40deb-6f144cb1f901
```

Hasilnya berupa tanda tangan sebagai berikut

30450221009ee39686b437f8d2f57e141192653d84481ba86f2fba-2dae0416d756f34477bd022039c225979716397d37cd77c3ea6a1abd-75fe43c2d736bcc7f40deb6f144cb1f901

- Membuat signed transaction

```
C:\BX>bx input-set “[ 30450221009ee39686b437f8d-2f57e141192653d84481ba86f2fba2dae0416d-756f34477bd022039c225979716397d37cd77c3ea6a1abd-75fe43c2d736bcc7f40deb6f144cb1f901 ] [ 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f-6019936f7e282969ee9a4f77 ]” 01000000128f093cd-c8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d-444ba910000000000ffffffff0200000000000000000506a4c-4d42656c616a6172206d656e797573756e207472616e73616b-7369204269746636f696e20736563617261206d-616e75616c2064656e67616e206d656e6767756e-616b616e2042582e2044696d617ae8fd000000000001976a91
```

```
4af4775b9355b39a7b2798bcdba1618ae89b-
679c688ac00000000
01000000012 8f093cdc8281930169eb4f08b-
96fe083a1e677373d4fd7133a6c9f3d444ba-
9100000000 6b483045022100 9ee39686b437f8d-
2f57e141192653d84481ba86f2fba2dae0416d756f34477b-
d022039c225979716397d37cd77c3ea6a1abd75fe43c2d-
736bcc7f40deb6f144cb1f90121034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77ffffff-
f0200000000000000000506a4c4d42656c616a6172206d656
e797573756e207472616e73616b73692042697463 6f696e-
2073656361726 1206d616e75616c2064656e67616e206d656e-
6767756e616b616e2042582e2044696d617ae8f-
d000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac00000000
```

Hasil akhir transaksinya yaitu:

```
0100000001 28f093cdc8281930169eb4f08b-
96fe083a1e677373d4fd7133a6c9f3d444ba-
91000000006b483045022100 9ee39686b437f8d-
2f57e141192653d84481ba86f2fba2dae0416d756f34477b-
d022039c225979716397d37cd77c3ea6a1abd75fe43c2d-
736bcc7f40deb6f144cb1f90121034b2a4e2dae1139794ef-
b11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77ff-
ffffff020000000000000000506a4c4d42656c616a6172206d
656e797573756e207472616e73616b73692042697463 6f696e-
2073656361726 1206d616e75616c2064656e67616e206d656e-
6767756e616b616e2042582e2044696d617ae8f-
d000000000001976a914af4775b9355b39a7b2798bcd-
ba1618ae89b679c688ac00000000
```


dapat dikirim ke jaringan Bitcoin.

- Pengiriman transaksi ke jaringan

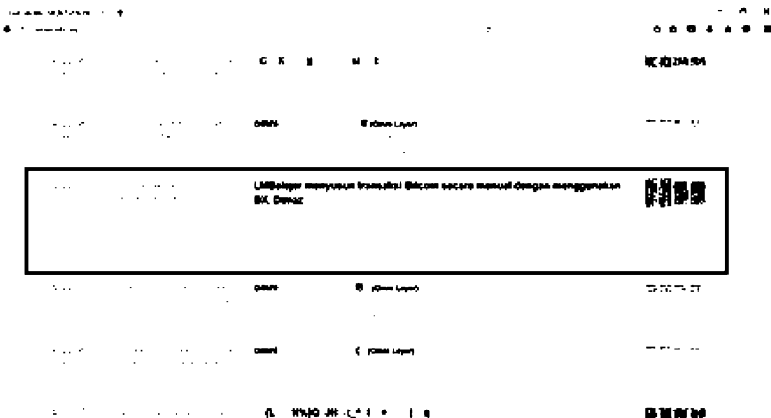
```
C:\BX>bx send-tx 01000000128f093cdc8281930169eb-
4f08b96fe083a1e677373d4fd7133a6c9f3d444ba-
91000000006b483045022100 9ee39686b437f8d-
2f57e141192653d84481ba86f2fba2dae0416d756f34477bd-
022039c225979716397d37cd77c3ea6a1abd75fe43c2d
```

```
736bcc7f40deb6f144cb1f90121034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77ffffff-
f0200000000000000506a4c4d42656c616a6172206d656e7-
97573756e207472616e73616b7369204269746636f696e-
2073656361617261206d616e75616c2064656e67616e206d-
656e6767756e616b616e2042582e2044696d-
617ae8fd000000000001976a914af4775b9355b39a7b2798b-
cdba1618ae89b679c688ac00000000
Sent transaction at 2016-Mar-12 11:26:55.
```

TxID atas transaksi di atas yaitu

`0975f444a42ee589dbddf7fc08763879a8fb645164a0da6f-c1afb7937fc740c1`

Apabila dikonfirmasi, transaksi null script akan tampil di situs Coinsecrets.org.



Gambar 26. Null Script dalam Coinsecrets.org

Pay To Script Hash Scripting

Pay To Script Hash (P2SH) merupakan tipe pembayaran Bitcoin yang dapat digunakan untuk berbagai keperluan, di antaranya multisignature dan hash-locked transaction. Berbagai model script dapat dibuat dengan menggunakan P2SH, tidak terbatas pada standar transaksi yang sudah ada. Artinya, pengguna Bitcoin dapat memanfaatkan P2SH untuk membuat transaksi kustom.

P2SH merupakan salah satu fitur unggulan Bitcoin, di mana berbagai kondisi dapat diciptakan di dalam P2SH sebagai syarat pembayaran. Misalnya, pembayar harus memiliki password yang tepat selain juga digital signature yang sah untuk dapat menerima pembayaran. Sebuah skema P2SH terdiri atas 2 fase, yakni commit dan redeem. Transaksi P2SH menggunakan alamat perantara yang disebut dengan P2SH address (alamat P2SH).

Multisignature

Multisignature sebagaimana telah dijelaskan pada bagian 2.2.7 dapat dibuat dengan menggunakan setidaknya 2 alamat Bitcoin. Terdapat banyak skema multisignature yang dapat dibuat, salah satu yang paling populer adalah 2-of-3 multisignature, yaitu skema yang membutuhkan 2 dari 3 signature. Skema 2-of-3 multisignature akan dibahas dalam bagian ini.

Skema multisignature ini akan dibedakan menjadi 2 bagian besar, masing-masing memiliki urutan langkah yang harus diselesaikan.

1. Multisignature: Commit

Fase commit merupakan fase di mana 1 atau lebih alamat asal mengirimkan uang kepada alamat P2SH dalam sebuah transaksi. Pada contoh berikut, uang yang dikirimkan berasal dari 3 alamat. Langkah-langkahnya sebagai berikut.

- Persiapan data

Skema 2-of-3 multisignature membutuhkan 3 pengirim, meskipun nantinya hanya dibutuhkan minimal 2 digital signature milik para pengirim tersebut.

```
                              Pengirim 1
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa

                              Pengirim 2
Private key: c8daee2ac304d64d9fe77ee30b7b092e611ea4d314d54727e14c3ba4dbf4ec24
Public key : 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77
ScriptPK   : dup hash160 [ af4775b9355b39a7b2798bcdba1618ae89b679c6 ] equalverify checksig
Address    : 1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr

                              Pengirim 3
Private key: bc9bec66f7eaff9f383091aedde72b27667cc27760fc0cce09c8f6932f5fbc86
Public key : 03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9
ScriptPK   : dup hash160 [ 7d150b6f24d21cc85114cee007b745b473e877c0 ] equalverify checksig
Address    : 1CQNgBtYwD8qHthBvSXrbtrFnVcX1TBXrb

                               Penerima
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ
```

Sementara itu, unspent transaction sebagai berikut

TxID Index Balance Pemilik

87a2b5474089ec7cf57ae5b492d107c848247951f7641ff2018b20937f0dfeda	0	20000	Pengirim 1
0975f444a42ee589dbddf7fc08763879a8fb645164a0da6fc1afb7937fc740c1	1	65000	Pengirim 2
91ba44d4f3c9a63371fdd47373671e3a08fe968bf0b49e16301928c8cd93f028	1	5000	Pengirim 3

- Menentukan P2SH script.

P2SH script merupakan ScriptPubKey yang harus dipenuhi agar mendapatkan pembayaran. P2SH script bergantung pada jenis transaksi P2SH yang dibuat. Dalam skema 2-of-3 multisignature, script yang dibutuhkan berbentuk sebagai berikut.

“2 <Pubkey 1> <Pubkey 2> <Pubkey 3> 3 OP_CHECKMULTISIG”

Di dalam perintah BX dan menggunakan data pada langkah (a), bentuknya menjadi

“2 [0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534] [034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77] [03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9] 3 checkmultisig”.

- Membuat alamat P2SH

Alamat P2SH dibuat berdasarkan P2SH script yang digunakan. Alamat P2SH dapat dibuat dengan menggunakan perintah “bx script-to-address <P2SH script>”

```
C:\BX>bx script-to-address "2 [
0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] [
034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77 ] [
03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9 ] 3
checkmultisig"
3PuipCXRtrGgeU5LxxZFCnMbYp2Y29kGZq
```

Alamat yang dihasilkan yakni 3PuipCXRtrGgeU5LxxZFCnMbYp2Y29kGZq merupakan alamat yang akan digunakan sebagai alamat tujuan bagi fase commit.

- Membuat unsigned transaction

Untuk unsigned transaction yang dibuat, input yang dibutuhkan berjumlah 3 yang berasal dari 3 unspent transaction dari langkah (a), sementara outputnya berjumlah 1 yaitu alamat P2SH **3PuipCXRtrGgeU5LxxZFCnMbYp2Y29kGZq.**

Dengan asumsi bahwa transaction fee sejumlah 10.000 satoshi, maka 80.000 satoshi dapat dikirimkan kepada alamat P2SH.

```
C:\BX>bx tx-encode -i 87a2b5474089ec7cf57ae5b492d107c848247951f7641ff2018b20937f0dfeda:0 -i 0975f444a42ee589dbddf7fc08763879a8fb645164a0da6fc1afb7937fc740c1:1 -i 91ba44d4f3c9a63371fdd47373671e3a08fe968bf0b49e16301928c8cd93f028:1 -o 3PuipCXRtrGgeU5LxxZFCnMbYp2Y29kGZq:80000
0100000003dafe0d7f93208b01f21f64f751792448c807d192b4e57af57cec894047b5a2870000000000ffffffffc140c77f93b7afc16fdaa0645164fba879387608fcf7dddb89e52ea444f4750901000000000ffffffff28f093cdc8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d444ba9101000000000ffffffff0180380100000000000017a914f3b9e47e4fca4ce91d481ec759e7917782bcac538700000000
```


- Menandatangani unsigned transaction.

Karena terdapat 3 buah input yang digunakan dalam unsigned transaction, maka diperlukan 3 buah digital signature, di mana masing-masing merupakan milik Pengirim 1, Pengirim 2, dan Pengirim 3.

Digital signature 1:

```
C:\BX>bx input-sign -i 0 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89 "dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig" 0100000003dafe0d7f93208b01f21f64f751792448c807d192b4e57af57cec894047b5a2870000000000ffffffffc140c77f93b7afc16fdaa0645164fba879387608fcf7dddb89e52ea444f4750901000000000ffffffff28f093cdc8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d444ba9101000000000ffffffff0180380100000000000017a914f3b9e47e4fca4ce91d481ec759e
```

```
7917782bcac538700000000
3045022100fdc178e951f75159636a6289eedaf-
0c35a750d2d08620bc1e64dbaa2f2f92f3a02200e175b-
b7ec03e9d63d5d1eeef87ec58469c0acc6013443834737d9ab-
235987fb01
```

Digital signature 2:

```
C:\BX>bx input-sign -i 1 c8daee2ac304d64d9fe77ee30b-
7b092e611ea4d314d54727e14c3ba4dbf4ec24 “dup hash160 [
af4775b9355b39a7b2798bcdba1618ae89b679c6 ] equalverify
checksig” 0100000003dafe0d7f93208b01f21f64f751792448c807d-
192b4e57af57cec894047b5a2870000000000ffffffffc140c77f-
93b7afc16fdaa0645164fba879387608fcf7dddb89e52-
ea444f47500100000000ffffffff28f093cdc8281930169eb4f08b-
96fe083a1e677373d4fd7133a6c9f3d444ba910100000000fffffff-
f0180380100000000000017a914f3b9e47e4fca4ce91d481ec759e-
7917782bcac538700000000
304402200f2a396537123f37c26de93f1c46339bd-
2b64ae54e1ab3e845190048f85ddf2302202e2c12e398238240c-
b12ab0bff2603b33f39946375953c4d0d8ef6f36d87b3b901
```

Digital signature 3:

```
C:\BX>bx input-sign -i 2 bc9bec66f7eaff9f383091aed-
de72b27667cc27760fc0cce09c8f6932f5fbc86 “dup hash160 [
7d150b6f24d21cc85114cee007b745b473e877c0 ] equalverify
checksig” 0100000003dafe0d7f93208b01f21f64f751792448c807d-
192b4e57af57cec894047b5a2870000000000ffffffffc140c77f-
93b7afc16fdaa0645164fba879387608fcf7dddb89e52-
ea444f47500100000000ffffffff28f093cdc8281930169eb4f08b-
96fe083a1e677373d4fd7133a6c9f3d444ba910100000000fffffff-
f0180380100000000000017a914f3b9e47e4fca4ce91d481ec759e-
7917782bcac538700000000
304402200aacbcf3488904f76e5547a3ac8c365d-
60c35af5eb42b37f9d1e6bd3d36e9a9b02201d0e91f000b-
8552c98264a340432f2884dac0157a56a26b778658a9c8037eb2901
```

- Menambahkan digital signature ke dalam unsigned transaction.

Proses penambahan digital signature atas transaksi dengan banyak input memiliki langkah-langkah yang serupa dengan bagian 4.10.2 yang telah dibahas, yakni penambahan secara bertahap, di mana hasil operasi yang satu akan digunakan dalam operasi berikutnya.

Penambahan digital signature 1:

```
C:\BX>bx input-set -i 0 "[ 3045022100fdc178e-
951f75159636a6289eedaf0c35a750d2d08620bc1e64d-
baa2f2f92f3a02200e175bb7ec03e9d63d5d1eeef87ec-
58469c0acc6013443834737d9ab235987fb01 ] [
0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534 ]" 0100000003dafe0d7f93208b01f21f64f-
751792448c807d192b4e57af57cec894047b5a2870000000000ffffffff-
c140c77f93b7afc16fdaa0645164fba879387608fcf7dddb89e52-
ea444f47509010000000ffffffff28f093cdc8281930169eb4f08b-
96fe083a1e677373d4fd7133a6c9f3d444ba910100000000ffffff-
f01803801000000000017a914f3b9e47e4fca4ce91d481ec759e-
7917782bcac538700000000

0100000003dafe0d7f93208b01f21f64f751792448c807d-
192b4e57af57cec894047b5a28700000000006b483045022100f-
dc178e951f75159636a6289eedaf0c35a750d2d08620b-
c1e64dbaa2f2f92f3a02200e175bb7ec03e9d63d5d1eeef87ec-
58469c0acc6013443834737d9ab235987fb01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534ffffffffc140c77f93b7afc16fdaa0645164f-
ba879387608fcf7dddb89e52ea444f47509010000000ffffffff28f-
093cdc8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d-
444ba910100000000ffffffff01803801000000000017a914f-
3b9e47e4fca4ce91d481ec759e7917782bcac538700000000
```

Penambahan digital signature 2:

```
C:\BX>bx input-set -i 1 "[ 304402200f2a396537123f37c26de93f-
1c46339bd2b64ae54e1ab3e845190048f85ddf2302202e2c12e-
398238240cb12ab0bff2603b33f39946375953c4d0d8ef6f36d87
```

```
b3b901 ] [ 034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77 ]"
0100000003dafe0d7f93208b01f21f64f751792448c807d-
192b4e57af57cec894047b5a287000000006b483045022100f-
dc178e951f75159636a6289eedaf0c35a750d2d08620bc1e-
64dbaa2f2f92f3a02200e175bb7ec03e9d63d5d1eeef87ec-
58469c0acc6013443834737d9ab235987fb01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8-
534ffffffffc140c77f93b7afc16fdaa0645164fba879387608f-
cf7dddb89e52ea444f47509010000000ffffffff28f093cd-
c8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d-
444ba910100000000ffffffff0180380100000000017a914f-
3b9e47e4fca4ce91d481ec759e7917782bcac538700000000
0100000003dafe0d7f93208b01f21f64f751792448c807d-
192b4e57af57cec894047b5a287000000006b483045022100f-
dc178e951f75159636a6289eedaf0c35a750d2d08620b-
c1e64dbaa2f2f92f3a02200e175bb7ec03e9d63d5d1eeef87ec-
58469c0acc6013443834737d9ab235987fb01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534ffffffffc140c77f93b7afc16fdaa0645164f-
ba879387608fcf7dddb89e52ea444f47509010000006a-
47304402200f2a396537123f37c26de93f1c46339bd-
2b64ae54e1ab3e845190048f85ddf2302202e2c12e398238240c-
b12ab0bff2603b33f39946375953c4d0d8ef6f36d87b3b-
90121034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f-
6019936f7e282969ee9a4f77ffffffff28f093cdc8281930169eb4f08b-
96fe083a1e677373d4fd7133a6c9f3d444ba910100000000fffffff-
f0180380100000000017a914f3b9e47e4fca4ce91d481ec759e-
7917782bcac538700000000
```

Penambahan digital signature 3:

```
C:\BX>bx input-set -i 2 "[ 304402200aacbcf-
3488904f76e5547a3ac8c365d60c35af5eb42b37f9d1e6bd-
3d36e9a9b02201d0e91f000b8552c98264a340432f-
2884dac0157a56a26b778658a9c8037eb2901 ] [
03608df49e57a4154dc84eeed2699da60c6452aa62be09dc-
76641582e2c2dbaed9 ]" 0100000003dafe0d7f93208b01f21f64f-
751792448c807d192b4e57af57cec894047b5a287000000006b483
```

045022100fdc178e951f75159636a6289eedaf-
0c35a750d2d08620bc1e64dbaa2f2f92f3a02200e175b-
b7ec03e9d63d5d1eeef87ec58469c0acc6013443834737d9ab-
235987fb01210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b8534ffffffffc140c77f-
93b7afc16fdaa0645164fba879387608fcf7dddb89e52e-
a444f47509010000006a47304402200f2a396537123f37c-
26de93f1c46339bd2b64ae54e1ab3e845190048f85ddf2302202e-
2c12e398238240cb12ab0bff2603b33f39946375953c4d-
0d8ef6f36d87b3b90121034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77ffffffff28f-
093cdc8281930169eb4f08b96fe083a1e677373d4fd7133a6c9f3d-
444ba91010000000ffffffff0180380100000000000017a914f-
3b9e47e4fca4ce91d481ec759e7917782bcac538700000000

0100000003dafe0d7f93208b01f21f64f751792448c807d-
192b4e57af57cec894047b5a287000000006b483045022100f-
dc178e951f75159636a6289eedaf0c35a750d2d08620b-
c1e64dbaa2f2f92f3a02200e175bb7ec03e9d63d5d1eeef87ec-
58469c0acc6013443834737d9ab235987fb01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534ffffffffc140c77f93b7afc16fdaa0645164f-
ba879387608fcf7dddb89e52ea444f47509010000006a-
47304402200f2a396537123f37c26de93f1c46339bd-
2b64ae54e1ab3e845190048f85ddf2302202e2c12e-
398238240cb12ab0bff2603b33f39946375953c4d-
0d8ef6f36d87b3b90121034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77ffffffff-
28f093cdc8281930169eb4f08b96fe083a1e677373d4f-
d7133a6c9f3d444ba91010000006a47304402200aacbcf-
3488904f76e5547a3ac8c365d60c35af5eb42b37f9d1e6bd3d36e-
9a9b02201d0e91f000b8552c98264a340432f2884dac0157a56
a26b778658a9c8037eb29012103608df49e57a4154dc84eeed-
2699da60c6452aa62be09dc76641582e2c2dbaed9ffffffff018038-
0100000000000017a914f3b9e47e4fca4ce91d481ec759e7917782b-
cac538700000000

- Mengirim transaksi ke jaringan Bitcoin.

```
  C:\BX>bx            send-tx            0100000003dafe0d-
7f93208b01f21f64f751792448c807d192b4e57af57ce-
c894047b5a2870000000006b483045022100fdc178e-
951f75159636a6289eedaf0c35a750d2d08620bc1e-
64dbaa2f2f92f3a02200e175bb7ec03e9d63d5d1eeef87ec-
58469c0acc6013443834737d9ab235987fb01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8-
534ffffffffc140c77f93b7afc16fdaa0645164f-
ba879387608fcf7dddb89e52ea444f47509010000006a-
47304402200f2a396537123f37c26de93f1c46339bd-
2b64ae54e1ab3e845190048f85ddf2302202e2c12e-
398238240cb12ab0bff2603b33f3994637595 3c4d-
0d8ef6f36d87b3b90121034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77fff-
fffff28f093cdc8281930169eb4f08b96fe083a1e677373d4f-
d7133a6c9f3d444ba91010000006a47304402200aacbcf-
3488904f76e5547a3ac8c365d60c35af5eb42b37f9d1e6bd-
3d36e9a9b02201d0e91f000b8552c98264a340432f2884dac0
157a56a26b778658a9c8037eb29012103608df49e57a4154d-
c84eeed2699da60c6452aa62be09dc76641582e2c2d-
baed9ffffffff0180380100000000000017a914f3b9e47e4fca-
4ce91d481ec759e7917782bcac538700000000
  Sent transaction at 2016-Mar-12 22:57:54.
```

Transaksi tersebut memiliki TxID sebagai berikut.

6e55d27db502dac5dc644febf0aa9c25b7677f56ef204379866d-6c78bd45a102

TxID di atas akan digunakan dalam fase redeem selanjutnya, yaitu mengirim transaksi dari alamat P2SH ke alamat tujuan.

2. Multisignature: Redeem

Fase redeem merupakan fase di mana alamat P2SH mengirimkan uang kepada alamat tujuan. Fase ini merupakan eksekusi dari fase commit yang telah dilakukan sebelumnya. Langkah-langkahnya sebagai berikut.

- Persiapan Data.

```
                              Pengirim 1
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa

                              Pengirim 2
Private key: c8daee2ac304d64d9fe77ee30b7b092e611ea4d314d54727e14c3ba4dbf4ec24
Public key : 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77
ScriptPK   : dup hash160 [ af4775b9355b39a7b2798bcdba1618ae89b679c6 ] equalverify checksig
Address    : 1GynrgvHWJKJQwskZS2gRzMntEVneCj1sr

                              Pengirim 3
Private key: bc9bec66f7eaff9f383091aedde72b27667cc27760fc0cce09c8f6932f5fbc86
Public key : 03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9
ScriptPK   : dup hash160 [ 7d150b6f24d21cc85114cee007b745b473e877c0 ] equalverify checksig
Address    : 1CQNgBtYwD8qHthBvSXrbtrFnVcX1TBXrb

                                 P2SH
Address    : 3PuipCXRtrGgeU5LxxZFCnMbYp2Y29kGZq
ScriptPK   : 2 [ 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] [
             034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77 ] [
             03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9 ] 3
             checkmultisig

                               Penerima
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ
```

Unsigned transaction yang dimiliki oleh alamat P2SH adalah sebagai berikut.

6e55d27db502dac5dc644febf0aa9c25b7677f56ef204379866d6c78bd45a102	0	80000

- Membuat unsigned transaction.

Transaksi akan dilakukan dari alamat P2SH menggunakan unsigned transaction yang dimilikinya kepada alamat tujuan sebesar 70.000 satoshi, yang berasal dari sisa uang sebesar 80.000 satoshi dikurangi biaya transaksi sebesar 10.000 satoshi.

```
C:\BX>bx tx-encode -i 6e55d27db502dac5dc644febf0aa9c25b7677f56ef204379866d6c78bd45a102:0 -o 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ:70000
010000000102a145bd786c6d86794320ef567f67b7259caaf0eb4f64dcc5da02b57dd2556e0000000000ffffffff0170110100000000001976a914cb40e725211a17c27604ae05cab333f20902a59888ac00000000
```

- Menandatangani unsigned transaction.

Terdapat perbedaan ScriptPubKey untuk P2SH jika dibandingkan dengan metode P2A yang digunakan dalam penandatanganan tran-

saksi. ScriptPubKey ini merupakan script yang sama dengan script yang digunakan untuk membuat alamat P2SH dalam fase commit.

Karena menggunakan skema 2-of-3 multisignature, maka dibutuhkan setidaknya 2 digital signature di antara 3 pengirim uang. Sebagai contoh akan digunakan digital signature yang dibuat oleh Pengirim 1 dan Pengirim 2 (tidak menutup kemungkinan kombinasti lain semisal Pengirim 1 dan Pengirim 3 atau Pengirim 2 dan Pengirim 3).

Digital signature Pengirim 1:

```
C:\BX>bx input-sign 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89 “2 [ 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] [ 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77 ] [ 03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9 ] 3 checkmultisig” 01000000102a145bd786c6d86794320ef567f67b7259caaf0eb4f64dcc5da02b57dd2556e0000000000ffffffff01701101000000000001976a914cb40e725211a17c27604ae05cab333f20902a59888ac00000000
3045022100be57416185ebcd6baeb296a7967ed618ff1c63470e6fb3ae1ca2a14bc43351ad0220506e49c99203aca4eb242df6b7babf966647cb0479416e5ec8483a6b1ee1b63601
```

Digital signature Pengirim 2:

```
C:\BX>bx input-sign c8daee2ac304d64d9fe77ee30b7b092e611ea4d314d54727e14c3ba4dbf4ec24 “2 [ 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] [ 034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77 ] [ 03608df49e57a4154dc84eeed2699da60c6452aa62be09dc76641582e2c2dbaed9 ] 3 checkmultisig” 01000000102a145bd786c6d86794320ef567f67b7259caaf0eb4f64dcc5da02b57dd2556e0000000000ffffffff01701101000000000001976a914cb40e725211a17c27604ae05cab333f20902a59888ac00000000
3045022100fc5982fac92a0d39ff3b45dff40983a2a0ec204c35871ea2e4085082dd79137c022020873ad04e729998bc82d9090c09bc621044cb601db998d3aaf56d3649a4a55a01
```

- Pengkodean ScriptPubKey

Selain dibutuhkan digital signature, skema P2SH juga membutuhkan informasi lain, yakni ScriptPubKey itu sendiri yang direpresentasikan ke dalam kode tertentu.

Untuk membuat kode ini, digunakan perintah “bx script-encode <script>”.

```
C:\BX>bx script-encode “2 [ 0372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534 ] [ 034b2a4e2dae1139794efb-
11c9e8d1dd55d5b7676f6019936f7e282969ee9a4f77 ] [
03608df49e57a4154dc84eeed2699da60c6452aa62be09dc-
76641582e2c2dbaed9 ] 3 checkmultisig”
5221037 2a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b853421034b2a4e-
2dae1139794efb11c9e8d1dd55d5b7676f-
6019936f7e282969ee9a4f772103608df49e57a4154dc84eeed-
2699da60c6452aa62be09dc76641582e2c2dbaed953ae
```


- Menambahkan digital signature ke dalam unsigned transaction.

Meskipun terdapat 2 digital signature, proses penambahan digital signature ke dalam unsigned transaction dilakukan sekaligus, karena kedua digital signature tersebut dianggap sebagai 1 kesatuan.

Format ScriptSig skema 2-of-3 multisignature yang harus dibuat agar dapat melakukan transaksi sebagaimana telah dibahas dalam bab 2.2.7 adalah sebagai berikut.

```
0 <signature 1> <signature 2> <P2SH script>
```

```
C:\BX>bx input-set “zero [ 3045022100be57416185e-
bcd6baeb296a7967ed618ff1c63470e6fb3ae1ca2a14b-
c43351ad0220506e49c99203aca4eb242df6b7bab-
f966647cb0479416e5ec8483a6b1ee1b63601 ] [
3045022100fc5982fac92a0d39ff3b45dff40983a2a0ec-
204c35871ea2e4085082dd79137c022020873ad04e729998bc-
82d9090c09bc621044cb601db998d3aaf56d3649a4a55a01 ] [
52210372a523947fb9a160669b4b8bf905ab
```

```
29d1ab906aa27901a2cd44aee17d7b853421034b2a4e-
2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee-
9a4f772103608df49e57a4154dc84eeed2699da-
60c6452aa62be09dc76641582e2c2dbaed953ae            ]"
01000000102a145bd786c6d86794320ef567f67b-
7259caaf0eb4f64dcc5da02b57dd2556e0000000000ffffff-
f0170110100000000001976a914cb40e725211a17c27604ae05cab-
333f20902a59888ac00000000
010000000102a145bd786c6d86794320ef567f67b-
7259caaf0eb4f64dcc5da02b57dd2556e00000000fdfe-
0000483045022100be57416185ebcd6baeb296a7967ed618ff-
1c63470e6fb3ae1ca2a14bc43351ad0220506e49c99203a-
ca4eb242df6b7babf966647cb0479416e5ec8483a6b1ee1b-
6360148304502210 0fc5982fac92a0d39ff3b45dff40983a2a0e-
c204c35871ea2e4085082dd79137c022020873ad04e729998b-
c82d9090c09bc621044cb601db998d3aaf56d3649a4a55a-
014c6952210372a523947fb9a160669b4b8bf905ab-
29d1ab906aa27901a2cd44aee17d7b853421034b2a4e-
2dae1139794efb11c9e8d1dd55d5b7676f6019936f7e282969ee-
9a4f772103608df49e57a4154dc84eeed2699da-
60c6452aa62be09dc76641582e2c2dbaed953aeffffff-
f0170110100000000001976a914cb40e725211a17c27604ae-
05cab333f20902a59888ac00000000
```


- Mengirim transaksi ke jaringan Bitcoin.

Setelah signed transaction berhasil dibuat, maka dapat dikirim ke jaringan.

```
C:\BX>bx        send-tx        010000000102a145bd786c-
6d86794320ef567f67b7259caaf0eb4f64dcc5da02b57d-
d2556e00000000fdfe0000483045022100be57416185e-
bcd6baeb296a7967ed618ff1c63470e6fb3ae1ca2a14b-
c43351ad0220506e49c99203aca4eb242df6b7babf966647c-
b0479416e5ec8483a6b1ee1b6360148304502210 0fc5982fa-
c92a0d39ff3b45dff40983a2a0ec204c35871ea2e4085082d-
d79137c022020873ad04e729998bc82d9090c09bc621044cb601db-
998d3aaf56d3649a4a55a014c6952210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b-
853421034b2a4e2dae1139794efb11c9e8d1dd55d5b7676f-
6019936f7e282969ee9a4f772103608df49e57a4154dc84eeed-
2699da60c6452aa62be09dc76641582e2c2dbaed953aeffffff-
f0170110100000000001976a914cb40e725211a17c27604ae05cab-
333f20902a59888ac00000000
Sent transaction at 2016-Mar-13 06:01:47.
```

Setelah menunggu beberapa saat, maka transaksi di atas yang memiliki TxID berikut akan terkonfirmasi.

```
29af74aac1f8f84d4db7bc1480d5024db06901a75da1018f-
b647ab37f5412f2f
```

Proses multisignature selesai.

Hash-Locked Transaction

Sebagaimana telah dibahas pada bab 2.2.8 tentang hash-locked transaction (HLT), terdapat 2 komponen dalam HLT, yakni password beserta nilai hash-nya, dan juga digital signature. Kedua komponen ini dibutuhkan untuk dapat mengambil pembayaran (redeem) dari pembayar. HLT dapat disusun dengan cara apapun sepanjang terdapat 2 komponen yang telah dijelaskan di atas.

Sama seperti P2SH lainnya, terdapat 2 fase dalam transaksi ini, yakni fase commit dan fase redeem. Fase commit merupakan fase di mana pembayar melakukan pembayaran kepada alamat P2SH. Sementara fase redeem merupakan fase di mana penerima pembayaran mengambil pembayaran dari alamat P2SH dengan menyertakan password dan digital signature yang tepat. Password ini diperoleh penerima dari si pengirim pembayaran.

Sebagai contoh, skema berikut merupakan salah satu HLT dalam fase commit.

```
OP_SHA256 <hash> OP_EQUALVERIFY <PubKey penerima>
OP_CHECKSIG
```

Skema di atas menggunakan operasi SHA256 untuk menghitung hash dari password yang digunakan.

Fase redeem memiliki format sebagai berikut.

```
<Signature Penerima> <Password> <P2SH script>
```

1. Hash-Locked Transaction: Commit

Langkah-langkah dalam fase ini dapat dijelaskan sebagai berikut.

- Persiapan Data.

```
                                Pengirim 1
Private key: 173250955a216d8dcb7f23840876dd1730c3ca2e1a73eb5cb88da4b30b6e0ea9
Public key : 034ea510ff96f20cd59d39e7bb6f488b12912630ee961f07e0011738994a4fdd4d
ScriptPK   : dup hash160 [ cb40e725211a17c27604ae05cab333f20902a598 ] equalverify checksig
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ

                                 Penerima
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
```

Sementara unspent transaction sebagai berikut.

29af74aac1f8f84d4db7bc1480d5024db06901a75da1018fb647ab37f5412f2f	0	70000

HLT membutuhkan password. Password ini dapat berupa apa saja, namun untuk lebih mudahnya digunakan hasil perintah seed dari toolkit BX dengan perintah **“`bx seed`”.**

```
C:\BX>bx seed
9e4d6b85de1fee6a54a1bd2ed4dc0642
```

Karena BX hanya dapat memproses informasi yang berformat Base16, maka password di atas harus diubah ke dalam Base16 dengan perintah **“`bx base16-encode <data>`”.**

```
C:\BX>bx base16-encode 9e4d6b85de1fee6a54a1bd2ed4dc0642
396534643662383564653166656536613534613162643265643464
6330363432
```

Nilai SHA256 atas password berformat Base16 di atas dapat dihitung dengan perintah “bx sha256 <data>”.

```
C:\BX>bx sha256
3965346436623835646531666565366135346131626432656434
646330363432
08a7f90481efb8c38e274b4c859b3eaef7783de956e-
6758770a68382c857712d
```

- Menentukan P2SH script.

Setelah informasi lengkap, maka menggunakan format berikut:

```
OP_SHA256 <hash> OP_EQUALVERIFY <PubKey penerima>
OP_CHECKSIG
```

P2SH script dapat disusun sebagai berikut.

```
OP_SHA256 08a7f90481efb8c38e274b4c859b3eaef-7783de956e6758770a68382c857712d OP_EQUALVERIFY 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2c-d44aee17d7b8534 OP_CHECKSIG
```

Script di atas dapat diubah menjadi script dalam BX menjadi sebagai berikut.

```
sha256 [ 08a7f90481efb8c38e274b4c859b3eaef7783de956e-6758770a68382c857712d ] equalverify [ 0372a523947f-b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] checksig
```


- Membuat alamat P2SH.

Untuk membuat alamat P2SH dari P2SH script, perintahnya adalah "bx script-to-address <script>".

```
C:\BX> bx script-to-address "sha256 [ 08a7f90481ef-b8c38e274b4c859b3eaef7783de956e6758770a68382c857712d ] equalverify [ 0372a523947fb9a160669b4b8bf905ab-29d1ab906aa27901a2cd44aee17d7b8534 ] checksig"
35xRinAvoed3QMSoJT1WxrbKiRiGouTKes
```

- Membuat unsigned transaction.

```
C:\BX> bx tx-encode -i 29af74aac1f8f84d4db7bc1480d5024d-b06901a75da1018fb647ab37f5412f2f:0 -o 35xRinAvoed3QMSo-JT1WxrbKiRiGouTKes:60000
01000000012f2f41f537ab47b68f01a15da70169b04d02d58014b-cb74d4df8f8c1aa74af290000000000ffffffff-f0160ea00000000000017a9142eca9b81247d8e85adb76bd-fa678629efdc3a5738700000000
```

- Menandatangani unsigned transaction.

```
C:\BX>bx input-sign 173250955a216d8dcb7f23840876d-d1730c3ca2e1a73eb5cb88da4b30b6e0ea9 “dup hash160 [ cb40e725211a17c27604ae05cab333f20902a598 ] equalverify checksig” 01000000012f2f-41f537ab47b68f01a15da70169b04d02d58014b-cb74d4df8f8c1aa74af2900000000000ffffff-f0160ea000000000000017a9142eca9b81247d8e85adb76bdfa6786-29efdc3a5738700000000
30440220249c5ae63dcabc0c9ab76b685cb1b4715898414d-6c78029afd1b975e52d01fb402205321c7721789e55f78ce-3239864895da6f6868c46dbb7024de542f4c07726ec501
```

- Menambahkan digital signature ke dalam unsigned transaction.

```
C:\BX>bx input-set “[ 30440220249c5ae63d-cabc0c9ab76b685cb1b4715898414d6c78029afd-1b975e52d01fb402205321c7721789e55f78ce-3239864895da6f6868c46dbb7024de542f4c07726ec501 ] [ 034ea510ff96f20cd59d39e7bb6f488b12912630ee961f-07e0011738994a4fdd4d ]” 01000000012f2f-41f537ab47b68f01a15da70169b04d02d58014b-cb74d4df8f8c1aa74af2900000000000ffffff-f0160ea000000000000017a9142eca9b81247d8e85adb76bd-fa678629efdc3a5738700000000
01000000012f2f41f537ab47b68f01a15d-a70169b04d02d58014bcb74d4df8f8c1aa74af29000000006a-4730440220249c5ae63dcabc0c9ab76b685cb1b4715898414d-6c78029afd1b975e52d01fb402205321c7721789e55f-78ce3239864895da6f6868c46dbb7024de542f-4c07726ec50121034ea510ff96f20cd59d39e7bb6f-488b12912630ee961f07e0011738994a4fdd4dffffff-f0160ea000000000000017a9142eca9b81247d8e85adb76bd-fa678629efdc3a5738700000000
```

dimaz ankaa wijaya

- Mengirim transaksi ke jaringan Bitcoin.

```
  C:\BX>bx     send-tx     01000000012f2f41f537ab47b-
68f01a15da70169b04d02d58014bcb74d4df8f8c1aa74a-
f2900000006a47304402202 49c5ae63dcabc0c9ab-
76b685cb1b4715898414d6c78029afd1b975e52d01fb-
402205321c7721789e55f78ce3239864895da6f6868c46dbb-
7024de542f4c07726ec50121034ea510ff96f20cd59d39e7bb-
6f488b12912630ee961f07e0011738994a4fdd4dffffff-
f0160ea00000000000017a9142eca9b81247d8e85adb76bd-
fa678629efdc3a57387000000000
  Sent transaction at 2016-Mar-21 11:23:15.
```

Transaksi di atas memiliki TxID sebagai berikut.

```
0e31e30ba8dbda82d0279bb21fa4fe745df6b9886bd23ea98ac-
094b0c975d375.
```

2. Hash-Locked Transaction: Redeem

Tidak seperti skema multisignature di mana fase redeem dikerjakan oleh para pengirim, fase redeem di dalam skema HLT dikerjakan oleh penerima. Untuk dapat menerima pembayaran, penerima harus mendapatkan password dari pengirim pembayaran dan juga menandatangani transaksi redeem. Jadi, dalam fase redeem, penerima harus memiliki digital signature dan password yang tepat.

Langkah-langkah dalam fase redeem dapat dirinci sebagai berikut.

- Persiapan Data.

Data yang dibutuhkan tidak jauh berbeda dengan fase commit dengan informasi alamat P2SH, P2SH script, dan password yang dibutuhkan.

```
                                    Pengirim 1
Private key: 173250955a216d8dcb7f23840876dd1730c3ca2e1a73eb5cb88da4b30b6e0ea9
Public key : 034ea510ff96f20cd59d39e7bb6f488b12912630ee961f07e0011738994a4fdd4d
ScriptPK   : dup hash160 [ cb40e725211a17c27604ae05cab333f20902a598 ] equalverify checksig
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ

                                       P2SH
Address    : 35xRinAvoed3QMSoJT1WxrbKiRiGouTKes
ScriptPK   :  sha256 [  08a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d  ]
              equalverify                                                                 [
              0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534          ]
              checksig
Password   : 9e4d6b85de1fee6a54a1bd2ed4dc0642
BASE16     : 39653464366238356465316665653661353461316264326564346463303634 32
```

```
                              Penerima
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
```

Unsigned transaction yang dimiliki oleh alamat P2SH adalah sebagai berikut.

[redacted]		
d856208a769115c56fcbcf7d774596d982487c528812b84a02e4bf77696e4849	0	50000

- Membuat unsigned transaction.

```
C:\BX>bx tx-encode -i d856208a769115c56fcbcf7d774596d982487c528812b84a02e4bf77696e4849:0 -o 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa:40000
01000000149486e6977bfe4024ab81288527c4882d99645777dcfcb6fc51591768a2056d80000000000ffffffff01409c00000000000001976a9142d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
```

- Menandatangani unsigned transaction.

```
C:\BX>bx input-sign 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89 “sha256 [ 08a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d ] equalverify [ 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] checksig” 01000000149486e6977bfe4024ab81288527c4882d99645777dcfcb6fc51591768a2056d80000000000ffffffff01409c00000000000001976a9142d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
3045022100d637fb8f2bdf4f3af56193a07c58c6392f784d8c11a66ab60596c536e47c6cb8022022d9b4e0e79a9ff662d7b2d3164f767b0dd5ca2710f652fcbe530f6c5f1baea801
```

- Pengkodean ScriptPubKey.

```
C:\BX>bx script-encode “sha256 [ 08a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d ] equalverify [ 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534 ] checksig”
a82008a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d88210372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534ac
```

- Menambahkan digital signature ke dalam unsigned transaction.

Selain digital signature, password juga harus ditambahkan ke dalam unsigned transaction dengan format sebagai berikut.

```
<Digital signature> <Password> <P2SH script>
```

```
C:\BX> bx input-set “[ 3045022100d637fb8f2bdf4f3af56193a07c58c6392f784d8c11a66ab60596c536e47c6cb8022022d9b4e0e79a9ff662d7b2d3164f767b0dd5ca2710f652fcbe530f6c5f1baea801 ] [ 39653464366238356465316665653661353461316264326564346463303634 32 ] [ a82008a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d88210372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534ac ]” 0100000001 49486e6977bfe4024ab81288527c4882d99645777dcfcb6fc51591768a2056d80000000000ffffffff01409c0000000000001976a9142d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
010000000149486e6977bfe4024ab81288527c4882d99645777dcfcb6fc51591768a2056d800000000b1483045022100d637fb8f2bdf4f3af56193a07c58c6392f784d8c11a66ab60596c536e47c6cb8022022d9b4e0e79a9ff662d7b2d3164f767b0dd5ca2710f652fcbe530f6c5f1baea8012039653464366238356465316665653661353461316264326564346463303634324 6a82008a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d88210372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534acffffffff01409c0000000000001976a9142d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
```


- Mengirim transaksi ke jaringan Bitcoin.

```
C:\BX>bx send-tx 010000000149486e6977bfe4024ab81288527c4882d99645777dcfcb6fc51591768a2056d800000000b1483045022100d637fb8f2bdf4f3af56193a07c58c6392f784d8c11a66ab60596c536e47c6cb8022022d9b4e0e79a9ff662d7b2d3164f767b0dd5ca2710f652fcbe530f6c5f1baea801203965346436623835646531666565366135346131626432656434646330363432 46a82008a7f90481efb8c38e274b4c859b3eaef7783de956e6758770a68382c857712d88210372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534acffffffff01409c0000000000001976a9142d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac00000000
Sent transaction at 2016-Mar-22 13:49:42.
```

Transaksi di atas memiliki TxID sebagai berikut.

69f8222e29dea81f2cb192dc93934126cad34e2a503294aee4256 9a981176185

Atomic Cross Chain Trading

Atomic Cross Chain Trading (ACCT) merupakan sebuah konsep yang dikembangkan oleh Ian Knowles dari Ciyam . ACCT merupakan konsep perdagangan antarcryptocurrency yang berbeda tanpa adanya pihak perantara. Dengan ACCT, 2 pihak yang ingin bertukar cryptocurrency yang berbeda dapat melakukannya secara langsung tanpa bantuan pihak ketiga. ACCT sedang berada dalam tahap pengembangan dan diharapkan dapat diintegrasikan di dalam platform Burst dan Qora.

ACCT memanfaatkan beberapa fitur dalam sistem Bitcoin yakni P2SH, HLT, LockTime, dan CLTV. P2SH digunakan untuk memberi dukungan terhadap script terkustomisasi, sementara HLT dimanfaatkan untuk menambah keamanan terhadap transaksi yang dibuat, yakni untuk memastikan identitas pihak penerima. Sementara itu CLTV digunakan untuk mengunci dana sampai pada waktu tertentu.

Script ACCT bekerja sebagai berikut. Pertama ia mengecek apakah penerima mampu menyebutkan password yang cocok dengan nilai hash tertentu. Apabila password tersebut cocok, maka penerima dapat mengambil dana yang dibayarkan dengan memberikan digital signature yang sesuai. Apabila dana tersebut tidak diambil oleh pihak penerima sampai pada jangka waktu tertentu, maka dana dapat diambil kembali oleh pihak pengirim.

Script yang digunakan dalam ACCT dapat berbentuk sebagai berikut.

```
OP_DUP OP_SHA256 <HASH> OP_EQUAL
OP_IF
      OP_DROP
      OP_DUP OP_HASH160 <pubkeyhash penerima> OP_EQUALVERIFY OP_CHECKSIG
OP_ELSE
      <CLTV>
      OP_NOP2
      OP_DROP
      OP_DUP OP_HASH160 <pubkeyhash pengirim> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

Dana yang dikirim dengan menggunakan script ACCT di atas dapat diambil dengan menggunakan 2 cara. Cara pertama dilakukan den-

gan menggunakan format berikut.

<digital signature penerima> <pubkey penerima> <pass-word> <P2SH script>

Sementara cara kedua dilakukan dengan menggunakan format di bawah ini.

<digital signature pengirim> <pubkey pengirim> <P2SH script>

Perbedaannya adalah, cara pertama digunakan untuk kasus normal di mana penerima berhasil mengklaim uang yang telah dibayarkan oleh pihak pengirim. Sementara cara kedua dilakukan jika transaksi dibatalkan atau pihak penerima tidak dapat memberikan password hingga batas waktu yang telah ditentukan.

Seperti halnya transaksi P2SH lain, terdapat 2 fase yang harus dilalui dalam skema ACCT, yakni fase commit dan redeem.

1. ACCT: Commit

Fase commit memiliki langkah-langkah sebagai berikut.

- Persiapan Data.

```
                               Pengirim 1
Private key: 173250955a216d8dcb7f23840876dd1730c3ca2e1a73eb5cb88da4b30b6e0ea9
Public key : 034ea510ff96f20cd59d39e7bb6f488b12912630ee961f07e0011738994a4fdd4d
ScriptPK   : dup hash160 [ cb40e725211a17c27604ae05cab333f20902a598 ] equalverify checksig
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ

                                Penerima
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
```

Unspent transaction dari pengirim adalah sebagai berikut.

a7a278a0d37eefc7840718b32d97142cd1e02904dbd32eb3e852f0f2e9be3b39	0	30000
c9b718dfab62954e93faeec1bf5b44127b1df869dab93ab85e8c7582444696fa	0	30000

Contoh penyusunan transaksi berikut menggunakan unspent transaction pertama.

- Membuat P2SH Script.

Pertama-tama, harus ditentukan password yang digunakan. Sebagai contoh mudah, akan digunakan "1234567890" yang harus diubah ke dalam format BASE16.

```
C:\BX>bx base16-encode 1234567890
31323334353637383930
```

Setelah password ditentukan, maka harus dihitung nilai hashnya, misalnya dengan menggunakan fungsi SHA256.

```
C:\BX>bx sha256 31323334353637383930
c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f268646
```

Setelah itu, untuk menentukan batas CLTV, harus diketahui nilai blok terakhir.

```
C:\BX>bx fetch-height
404058
```

Dengan nilai blok terakhir 404058, maka nilai CLTV dapat dihitung dengan cara menambahkan nilai blok terakhir dengan jangka selisih waktu yang diinginkan, misalnya diperlukan waktu kira-kira 2 jam atau 120 menit, maka waktu ini dapat dikonversi ke dalam panjang blok menjadi 12 blok dengan asumsi bahwa setiap blok baru memerlukan waktu 10 menit. Oleh karena itu nilai CLTV adalah 404058 + 12 = 404070.

Menggunakan bantuan calculator for programmer, nilai CLTV ini harus diubah ke dalam bentuk heksadesimal menjadi 62a66. Untuk kepentingan pengubahan ke dalam little endian, diperlukan angka berjumlah genap, oleh karena itu nilai heksadesimal tersebut dapat diubah menjadi 062a66 tanpa mengubah nilainya. Nilai ini kemudian diubah ke dalam format little endian menjadi 662a06.

Jika nilai-nilai ini telah didapatkan, maka script P2SH dapat dibuat.

```
"dup sha256 [ c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f268646 ] equal if drop dup hash160
[ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify
checksig else [ 662a06 ] nop2 drop dup hash160 [ cb40e725211a-
17c27604ae05cab333f20902a598 ] equalverify checksig endif"
```

Script tersebut kemudian diubah menjadi kode P2SH.

```
  C:\BX>bx script-encode “dup sha256 [ c775e7b757ed-
e630cd0aa1113bd102661ab38829ca52a6422ab782862f268646 ]
equal if drop dup hash160 [ 2d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe ] equalverify checksig else [ 662a06
] nop2 drop dup hash160 [ cb40e725211a17c27604ae05cab-
333f20902a598 ] equalverify checksig endif”
  76a820c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f26864687637576a9142d814b-
6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac6703662a-
06b17576a914cb40e725211a17c27604ae05cab-
333f20902a59888ac68
```

- Membuat alamat P2SH.

```
  C:\BX>bx script-to-address “dup sha256 [ c775e7b757e-
de630cd0aa1113bd102661ab38829ca52a6422ab782862f268646
] equal if drop dup hash160 [ 2d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe ] equalverify checksig else [ 662a06
] nop2 drop dup hash160 [ cb40e725211a17c27604ae05cab-
333f20902a598 ] equalverify checksig endif”
  3DeHymHmVXk5p2AbsJ9P4ZSQC4yYioxnrc
```


- Membuat unsigned transaction.

```
  C:\BX> bx tx-encode -i a7a278a0d37eefc7840718b32d-
97142cd1e02904dbd32eb3e852f0f2e9be3b39:0:1 -o 3DeHymHm-
VXk5p2AbsJ9P4ZSQC4yYioxnrc:20000 -l 404063
  0100000001393bbee9f2f052e8b32ed3db0429e0d12c14972d-
b3180784c7ef7ed3a078a2a70000000000010000001204e000000
00000017a914831da9fbee8ca23e08f0b42f9249c3f32583ac9b-
875f2a0600
```

- Menandatangani unsigned transaction.

```
  C:\BX> bx input-sign 173250955a216d8dcb7f23840876d-
d1730c3ca2e1a73eb5cb88da4b30b6e0ea9    “dup    hash160
[      cb40e725211a17c27604ae05cab333f20902a598      ]
equalverify      checksig”      0100000001393bbee9f-
2f052e8b32ed3db0429e0d12c14972db3180784c7ef7ed-
3a078a2a70000000000010000001204e00000000000017a914831
```

```
  da9fbee8ca23e08f0b42f9249c3f32583ac9b875f2a0600
  3044022028cdc2dbc5bdef5221d191ba1fde56c07b-
d7294a742951d71cfb9439b9931df4022029883e1ae2f5d2e4c7d-
785228b634188e33ccb77645d4303988dd8fb140c4b0201
```

- Menambahkan digital signature ke dalam unsigned transaction.

```
  C:\BX>bx      input-set     “[      3044022028cdc2d-
bc5bdef5221d191ba1fde56c07bd7294a742951d-
71cfb9439b9931df4022029883e1ae2f5d2e4c7d-
785228b634188e33ccb77645d4303988dd8fb140c4b0201       ]
[       034ea510ff96f20cd59d39e7bb6f488b12912630ee961f-
07e0011738994a4fdd4d       ]”      0100000001393bbee9f-
2f052e8b32ed3db0429e0d12c14972db3180784c7ef7ed-
3a078a2a7000000000000100000001204e00000000000017a914831
da9fbee8ca23e08f0b42f9249c3f32583ac9b875f2a0600
  0100000001393bbee9f2f052e8b32ed3db0429e0d12c14972d-
b3180784c7ef7ed3a078a2a7000000006a473044022028cd-
c2dbc5bdef5221d191ba1fde56c07bd7294a742951d71cfb9439-
b9931df4022029883e1ae2f5d2e4c7d785228b634188e33c-
cb77645d4303988dd8fb140c4b020121034ea510ff96f20cd-
59d39e7bb6f488b12912630ee961f07e0011738994a4fdd4d0100-
000001204e00000000000017a914831da9fbee8ca23e08f0b42f-
9249c3f32583ac9b875f2a0600
```

Transaksi di atas memiliki TxID sebagai berikut.

c1933ac8f31c8ff0f9f7f2b94aaf9dac7d66e-
f37656384307320375a3646601

- Mengirim transaksi ke jaringan Bitcoin.

```
  C:\BX>bx         send-tx         0100000001393bbee9f-
2f052e8b32ed3db0429e0d12c14972db3180784c7ef7ed-
3a078a2a7000000006a473044022028cdc2dbc5bdef5221d191ba1f-
de56c07bd7294a742951d71cfb9439b9931df4022029883e1ae2f-
5d2e4c7d785228b634188e33ccb77645d4303988dd8fb140c-
4b020121034ea510ff96f20cd59d39e7bb6f488b12912630ee961f-
07e0011738994a4fdd4d0100000001204e00000000000001
7a914831da9fbee8ca23e08f0b42f9249c3f32583ac9b875f2a0600
  Sent transaction at 2016-Mar-24 23:03:55.
```

1. ACCT: Redeem

Seperti telah dijelaskan sebelumnya, terdapat 2 cara untuk mengambil dana yang telah dikirimkan dalam fase commit. Cara pertama dilakukan oleh penerima dana, dan cara kedua dilakukan oleh pengirim dana apabila waktu CLTV telah berlalu.

a. ACCT Redeem Skema 1

Skema 1 membutuhkan keterlibatan pihak penerima untuk membuka informasi password selain juga digital signature. Pada skema ini sebenarnya tidak diperlukan sequence number dan LockTime, namun dapat digunakan juga untuk menambah fitur pengaturan waktu. Langkah-langkahnya adalah sebagai berikut.

- Persiapan data.

```
                              Pengirim 1
Private key: 173250955a216d8dcb7f23840876dd1730c3ca2e1a73eb5cb88da4b30b6e0ea9
Public key : 034ea510ff96f20cd59d39e7bb6f488b12912630ee961f07e0011738994a4fdd4d
ScriptPK   : dup hash160 [ cb40e725211a17c27604ae05cab333f20902a598 ] equalverify checksig
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ

                               Penerima
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
```


Sementara itu unspent transaction yang berasal dari fase commit sebagai berikut.

c1933ac8f31c8ff0f9f7f2b94aaf9dac7d66ef376566384307320375a3646601	0	20000

- Membuat unsigned transaction.

```
 C:\BX>bx   tx-encode   -i   c1933ac8f31c8ff0f9f7f-
2b94aaf9dac7d66ef376566384307320375a3646601:0:1   -o
159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa:10000 -l 404063
 010000000101666 4a3750332074338666537ef-
667dac9daf4ab9f2f7f9f08f1cf3c83a-
93c10000000000001000000011027000000000000001976a914
2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac5f2a0600
```

- Menandatangani unsigned transaction.

```
  C:\BX>bx    input-sign    23e9e2d5bc313418a9d5ca65ae-
ae181b0e77bfe400f79543494d703cb5a7be89              “dup
sha256      [      c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f268646 ] equal if drop dup hash160
[ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalver-
ify checksig else [ 662a06 ] nop2 drop dup hash160 [
cb40e725211a17c27604ae05cab333f20902a598 ] equalverify
checksig endif” 0100000001016664a375033207433866537ef-
6 6 7 d a c 9 d a f 4 a b 9 f 2 f 7 f 9 f 0 8 f 1 c f 3 c 8 3 a -
93c1000000000001000000011027000000000000001976a914
2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac5f2a0600
  3 0 4 5 0 2 2 1 0 0 c c c 8 2 5 2 b 9 5 9 e a 6 b e 5 9 2 c -
e365add19a4667f632f4b49770b42facd23b91e-
49496022014a13b11be47f5898a2d6df4538db6e2c6d-
5b83a447e542835489aad330738da01
```

- Menambahkan digital signature ke dalam unsigned transac-
  tion.

```
  C:\BX>bx   input-set   “[   3045022100ccc8252b959ea-
6be592ce365add19a4667f632f4b49770b42facd-
23b91e49496022014a13b11be47f5898a2d6df4538d-
b6e2c6d5b83a447e542835489aad330738da01         ]       [
0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2c-
d44aee17d7b8534    ]   [   313233343536373839 30   ]   [
76a820c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f26864687637576a9142d814b6ba-
2c3099aa5b3d38e2cacfa13d28aaebe88ac6703662a06b17576a91-
4cb40e725211a17c27604ae05cab333f20902a59888ac68
]”             0100000001016664a375033207433866537ef-
6 6 7 d a c 9 d a f 4 a b 9 f 2 f 7 f 9 f 0 8 f 1 c f 3 c 8 3 a -
93c1000000000001000000011027000000000000001976a914
2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe88ac5f2a0600
  0100000001016664a375033207433866537ef667dac9da-
f4ab9f2f7f9f08f1cf3c83a93c100000000d8483045022100c-
cc8252b959ea6be592ce365add19a4667f632f4b49770b-
42facd23b91e49496022014a13b11be47f5898a2d6df4538db-
6e2c6d5b83a447e542835489aad330738da01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d-
7b85340a3132333435363738393 04c6076a820c775e7b757e-
de630cd0aa1113bd102661ab38829ca52a6422ab-
782862f26864687637576a9142d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe88ac6703662a06b17576a914cb40e725211a-
17c27604ae05cab333f20902a59888ac6801000000011027000000
0000001976a9142d814b6ba2c3099aa5b3d38e2cacfa13d28aae-
be88ac5f2a0600
```

- Mengirim transaksi ke jaringan Bitcoin.

```
  C:\BX>bx                                    send-tx
0100000001016664a37503320743386665 37ef667dac9da-
f4ab9f2f7f9f08f1cf3c83a93c100000000d8483045022100c-
cc8252b959ea6be592ce365add19a4667f632f4b49770b42-
facd23b91e49496022014a13b11be47f5898a2d6df4538db-
6e2c6d5b83a447e542835489aad330738da01210372a523947f-
b9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8-
5340a313233343536373839304c6076a820c775e7b757e-
de630cd0aa1113bd102661ab38829ca52a6422ab-
782862f268646876375 76a9142d814b6ba2c3099aa5b-
3d38e2cacfa13d28aaebe88ac6703662a06b17576a914c-
b40e725211a17c27604ae05cab333f20902a59888ac680100000
001102700000000000001976a9142d814b6ba2c3099aa5b3d38e-
2cacfa13d28aaebe88ac5f2a0600
  Sent transaction at 2016-Mar-24 23:43:20.
```

Transaksi di atas dapat dilihat dengan TxID sebagai berikut.

e1733549442a05ce3e1f56cb0e52b4542ab793b3fb55cb-cf3848559fcb242c93

b. ACCT Redeem Skema 2

Skema kedua dilakukan jika pihak pengirim ingin mengklaim kembali dana yang telah dikirimkan tetapi tidak dapat diambil oleh pihak penerima hingga batas waktu tertentu yang telah ditentukan sebelumnya. Dalam menggunakan skema 2 ini perlu diperhatikan bahwa nilai LockTime harus sama dengan atau lebih besar dari nilai CLTV.

- Persiapan data.

```
                         Pengirim 1
Private key: 173250955a216d8dcb7f23840876dd1730c3ca2e1a73eb5cb88da4b30b6e0ea9
Public key : 034ea510ff96f20cd59d39e7bb6f488b12912630ee961f07e0011738994a4fdd4d
ScriptPK   : dup hash160 [ cb40e725211a17c27604ae05cab333f20902a598 ] equalverify checksig
Address    : 1KXhvUYN1VZpnJ9kQTkrW2mv7zQLiFMzwJ

                          Penerima
Private key: 23e9e2d5bc313418a9d5ca65aeae181b0e77bfe400f79543494d703cb5a7be89
Public key : 0372a523947fb9a160669b4b8bf905ab29d1ab906aa27901a2cd44aee17d7b8534
ScriptPK   : dup hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ] equalverify checksig
Address    : 159cJBQeWyLx9hP6hz3tYBc3mvHAzn8Goa
```

Sementara itu, untuk menjelaskan skema kedua ini dibuat unspent transaction yang berasal dari fase commit kedua dengan informasi sebagai berikut.

6ee9db9315ffa1664df8562d48b719f99d5d60486fd588e04eb6e2737229cbc2	0	20000

- Membuat unsigned transaction.

Sebagaimana telah diketahui bahwa CLTV bernilai 404070, maka LockTime dapat disetting bernilai 40471.

```
 C:\BX>bx tx-encode -i 6ee9db9315f-
fa1664df8562d48b719f99d5d60486fd588e04e-
b6e2737229cbc2:0:2 -o 1KXhvUYN1VZpnJ9kQTkrW2m-
v7zQLiFMzwJ:10000 -l 404071
 0100000001c2cb297273e-
2b64ee088d56f48605d9df919b7482d56f-
84d66a1ff1593dbe96e-
000000000200000001102700000000000001976a914c-
b40e725211a17c27604ae05cab333f20902a59888ac672a0600
```

- Menandatangani unsigned transaction.

```
 C:\BX>bx input-sign 173250955a216d8dcb7f23840876d-
d1730c3ca2e1a73eb5cb88da4b30b6e0ea9 “dup
sha256 [ c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f268646 ] equal if drop dup
hash160 [ 2d814b6ba2c3099aa5b3d38e2cacfa13d28aaebe ]
equalverify checksig else [ 662a06 ] nop2 drop dup
hash160 [ cb40e725211a17c27604ae05cab333f20902a598
] equalverify checksig endif” 0100000001c2cb297273e-
2b64ee088d56f48605d9df919b7482d56f84d66a1ff1593d-
be96e000000000200000001102700000000000001976a914c-
b40e725211a17c27604ae05cab333f20902a59888ac672a0600
 304502210083028edb7f469d1a5d558103b86a2923017eb-
5cc5b6abff86e6841e4a0139c8802207fe782181834422d-
4dc20e7a0cbe420265ca7e05be8463bc596c62b2e14f50ea01
```

- Menambahkan digital signature ke dalam unsigned transaction.

```
 C:\BX> bx input-set “[ 304502210083028edb7f-
469d1a5d558103b86a2923017eb5cc5b6abff86e6841e-
4a0139c8802207fe782181834422d4dc20e7a0cbe-
420265ca7e05be8463bc596c62b2e14f50ea01 ] [
034ea510ff96f20cd59d39e7bb6f488b12912630ee961f-
07e0011738994a4fdd4d ] [ 76a820c775e7b757e-
de630cd0aa1113bd102661ab38829ca52a6422ab-
782862f26864687637576a9142d814b6ba2c3099aa5b3d38e2cac-
fa13d28aaebe88ac6703662a06b17576a91
```

```
4cb40e725211a17c27604ae05cab-
333f20902a59888ac68        ]”        0100000001c2cb297273e-
2b64ee088d56f48605d9df919b7482d56f84d66a1ff1593d-
be96e0000000000200000001102700000000000001976a914c-
b40e725211a17c27604ae05cab333f20902a59888ac672a0600
0100000001c2cb297273e-
2b64ee088d56f48605d9df919b7482d56f84d66a1f-
f1593dbe96e00000000cd483045022100830228edb7f-
469d1a5d558103b86a2923017eb5cc5b6abff86e6841-
e4a0139c8802207fe782181834422d4dc20e7a0cbe-
420265ca7e05be8463bc596c62b2e14f50ea0121034e-
a510ff96f20cd59d39e7bb6f488b12912630ee961f-
07e0011738994a4fdd4d4c6076a820c775e7b757e-
de630cd0aa1113bd102661ab38829ca52a6422ab-
782862f26864687637576a9142d814b6ba2c3099aa5b-
3d38e2cacfa13d28aaebe88ac6703662a06b17576a914cb-
40e725211a17c27604ae05cab333f20902a59888a-
c680200000001102700000000000001976a914cb40e725211a-
17c27604ae05cab333f20902a59888ac672a0600
```

- Mengirim transaksi ke jaringan Bitcoin.

```
C:\BX>      bx      send-tx      0100000001c2cb297273e-
2b64ee088d56f48605d9df919b7482d56f84d66a1f-
f1593dbe96e00000000cd483045022100830228edb7f-
469d1a5d558103b86a2923017eb5cc5b6abff86e6841e-
4a0139c8802207fe782181834422d4dc20e7a0cbe420265ca7e-
05be8463bc596c62b2e14f50ea0121034ea510ff96f20cd-
59d39e7bb6f488b12912630ee961f07e0011738994a4fd-
d4d4c6076a820c775e7b757ede630cd0aa1113bd102661ab-
38829ca52a6422ab782862f26864687637576a9142d814b6ba-
2c3099aa5b3d38e2cacfa13d28aaebe88ac6703662a06b17576a-
914cb40e725211a17c27604ae05cab333f20902a59888a-
c680200000001102700000000000001976a914cb40e725211a-
17c27604ae05cab333f20902a59888ac672a0600
Sent transaction at 2016-Mar-25 05:16:30.
```

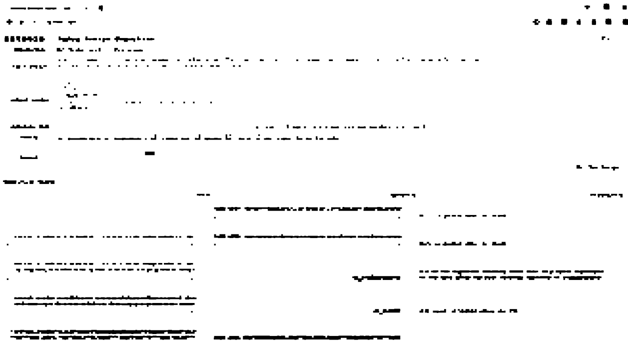
Transaksi di atas menghasilkan TxID sebagai berikut.

d225f11fa743be0b911d40d1e66352dca90fd247d2d3d16da9bd43d64d091008

Kustomisasi Script

Pengguna Bitcoin dapat membuat script kustom atas P2SH dengan leluasa. Untuk membantu evaluasi script, pengguna dapat memanfaatkan fasilitas yang disediakan oleh Webbtc.com di https://webbtc.

com/script. Fasilitas ini membantu pengguna untuk melakukan debugging (pengecekan) atas script dan menentukan apakah script yang dibuat dapat dieksekusi dengan baik atau tidak.



Gambar 27. Evaluasi script dari webbtc.com

Mengingat bahwa script yang salah dapat mengakibatkan hilangnya bitcoin, maka tidak ada salahnya pengguna berhati-hati dalam membuat script terkustomisasi. Ujicoba atas script yang dibuat dapat dilakukan dengan menggunakan bitcoin dalam jumlah kecil untuk meminimalisasi risiko.

Referensi

1. Nakamoto, S., Bitcoin: A peer-to-peer electronic cash system. 2008.
2. Merkle, R.C. Protocols for public key cryptosystems. in 1980 Symposium on Security and Privacy, IEEE Computer Society. 1980. IEEE.
3. Dai, W. B-money. 1998 [cited 2015 October 5, 2015]; Available from: http://www.weidai.com/bmoney.txt.
4. Bitcoin Wiki. Zero Knowledge Contingent Payment. 2011 May 7, 2014 [cited 2015 September 28, 2015]; Available from: https://en.bitcoin.it/wiki/Zero_Knowledge_Contingent_Payment.
5. Shamir, A., How to share a secret. Communications of the ACM, 1979. 22(11): p. 612-613.
6. Back, A., Hashcash-a denial of service counter-measure. 2002.
7. Bitcoin Wiki. Difficulty. 2010 13 July 2015 [cited 2016 29 March]; Available from: https://en.bitcoin.it/wiki/Difficulty.
8. Harding, D.A. Bitcoin Developer Guide. 2015 [cited 2016 12 January 2016]; Available from: https://bitcoin.org/en/developer-guide.
9. Franco, P., Understanding Bitcoin: Cryptography, engineering, and economics. 2015: John Wiley & Sons Ltd.
10. Bartlett, J., The Dark Net. 2014: The Random House.
11. Bohr, J. and M. Bashir. Who Uses Bitcoin? An exploration of the Bitcoin community. in Privacy, Security and Trust (PST), 2014 Twelfth Annual International Conference on. 2014.
12. Smyth, L. THE POLITICS OF BITCOIN. 2014; Available from: http://simulacrum.cc/2014/03/07/the-politics-of-bitcoin/.
13. Bitcoin Wiki. Controlled supply. 2011 16 January 2016 [cited 2016 20 January 2016]; Available from: https://en.bitcoin.it/wiki/Controlled_supply.
14. Bitcoin Forum. 21 Million Cap. 2011 [cited 2016 14 March]; Available from: https://bitcointalk.org/index.php?topic=3366.0.
15. Bitcoin Wiki. Scalability. 2011 21 October 2015 [cited 2016 24 March]; Available from: https://en.bitcoin.it/wiki/Scalability.
16. Bitcoin Wiki. Weaknesses. 2011 8 July 2015 [cited 2016 23 March]; Available from: https://en.bitcoin.it/wiki/Weaknesses.
17. Bitcoin Wiki. Transaction Malleability. 2013 18 August 2015

[cited 2016 24 March]; Available from: https://en.bitcoin.it/wiki/Transaction_Malleability.
18. Ron, D. and A. Shamir, Quantitative analysis of the full bitcoin transaction graph, in Financial Cryptography and Data Security. 2013, Springer. p. 6-24.
19. Laurie, B., Decentralised currencies are probably impossible (but let's at least make them efficient). Practice, 2011: p. 1-10.
20. Stinson, D.R., Cryptography Theory and Practice. 1995: CRC Press.
21. Diffie, W. and M.E. Hellman. Multiuser cryptographic techniques. in Proceedings of the June 7-10, 1976, national computer conference and exposition. 1976. ACM.
22. Rivest, R.L., A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 1978. 21(2): p. 120-126.
23. ElGamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. in Advances in cryptology. 1985. Springer.
24. Brown, D.R., Recommended Elliptic Curve Domain Parameters. Certicom Corp, 2010.
25. Bitcoin Wiki. RIPEMD-160. 2014 June 30, 2014; Available from: https://en.bitcoin.it/wiki/RIPEMD-160.
26. Yao, A.C.-C. Protocols for secure computations. in FOCS. 1982.
27. Andrychowicz, M., et al. Secure multiparty computations on bitcoin. in Security and Privacy (SP), 2014 IEEE Symposium on. 2014. IEEE.
28. Bitcoin Wiki. Base58Check encoding. 2011 July 3, 2015 [cited 2015 September 21, 2015]; Available from: https://en.bitcoin.it/wiki/Base58Check_encoding.
29. Bitcoin Wiki. Technical background of version 1 Bitcoin addresses. 2011 August 17, 2015 [cited 2015 September 21, 2015]; Available from: https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses.
30. Bitcoin Wiki. Script. 2013 September 25, 2015 [cited 2015 September 28, 2015]; Available from: https://en.bitcoin.it/wiki/Script.
31. Bitcoin Wiki. Transaction. 2013 28 May 2015 [cited 2016 19 January 2016]; Available from: https://en.bitcoin.it/wiki/Trans-

action.

32. Bitcoin Wiki. Pay to Script Hash. 2012 27 May 2015 [cited 2016 9 January 2016]; Available from: https://en.bitcoin.it/wiki/Pay_to_script_hash.
33. Andresen, G. Pay to Script Hash. 2012 [cited 2016 9 January 2016]; Available from: https://github.com/bitcoin/bips/blob/master/bip-0016.mediawiki.
34. Andresen, G. M-of-N Standard Transactions. 2011 [cited 2015 September 28, 2015]; Available from: https://github.com/bitcoin/bips/blob/master/bip-0011.mediawiki.
35. Tiernan, N. Hash Locked Transaction. 2014 April 26, 2014 [cited 2015 September 28, 2015]; Available from: https://github.com/TierNolan/bips/blob/bip4x/bip-0045.mediawiki.
36. Tiernan, N. Alt Chains and Atomic Transfers. 2013 May 7, 2013 [cited 2015 September 28, 2015]; Available from: https://bitcointalk.org/index.php?topic=193281.msg2224949#msg2224949.
37. Bitcoin Wiki. Contract. 2012 July 8, 2015 [cited 2015 September 28, 2015]; Available from: https://en.bitcoin.it/wiki/Contract.
38. Harding, D.A. Locktime, nLockTime. 2015 [cited 2016 12 January 2016]; Available from: https://bitcoin.org/en/glossary/locktime.
39. Harding, D.A. Sequence Number (Transactions). 2015 [cited 2016 12 January 2016]; Available from: https://bitcoin.org/en/glossary/sequence-number.
40. Todd, P. OP_CHECKLOCKTIMEVERIFY. 2014 [cited 2016 12 January 2016]; Available from: https://github.com/bitcoin/bips/blob/master/bip-0065.mediawiki.
41. Wikipedia. Endianness. 2015 20 February 2016 [cited 2016 13 March]; Available from: https://en.wikipedia.org/wiki/Endianness.
42. ThePiachu. Why does the Bitcoin protocol use the little-endian notation? 2011 [cited 2016 14 March]; Available from: http://bitcoin.stackexchange.com/questions/2063/why-does-the-bitcoin-protocol-use-the-little-endian-notation.
43. Bitcoin Forum. What would you change about the Bitcoin protocol? 2011 [cited 2016 14 March]; Available from: What would you change about the Bitcoin protocol?

44. Caetano, R., Learning Bitcoin. 2015: Packt Publishing.
45. Bitcoin Wiki. Transaction fees. 2011 11 December 2015 [cited 2016 27 January 2016]; Available from: https://en.bitcoin.it/wiki/Transaction_fees.
46. Dashjr, L. Restore minimum feerate to 10000 satoshis. 2015 [cited 2016 12 March]; Available from: https://github.com/bitcoin/bitcoin/pull/6201.
47. Bitcoin Wiki. Units. 2011 [cited 2016 14 March]; Available from: https://en.bitcoin.it/wiki/Units.
48. Maxwell, G. Deterministic Wallets. 2011 [cited 2015 September 12, 2015]; Available from: https://bitcointalk.org/index.php?topic=19137.0.
49. Wuille, P. Hierarchical Deterministic Wallets. 2012 [cited 2016 29 February]; Available from: https://github.com/bitcoin/bips/blob/master/bip-0032.mediawiki.
50. Castillo, M.D. Dark Wallet : A radical Way to Bitcoin. 2013 [cited 2015 October 8, 2015]; Available from: http://www.newyorker.com/currency-tag/dark-wallet-a-radical-way-to-bitcoin.
51. Todd, P. Stealth Addresses. 2014 [cited 2015 October 8, 2015]; Available from: http://sourceforge.net/p/bitcoin/mailman/message/31813471/.
52. unSYSTEM Wiki. DarkWallet/Stealth. 2014 November 27, 2014; Available from: https://wiki.unsystem.net/en/index.php/DarkWallet/Stealth.
53. US Department of Justice. Manhattan U.S. Attorney Announces Charges Against Liberty Reserve. 2013; Available from: http://www.justice.gov/usao-sdny/pr/manhattan-us-attorney-announces-charges-against-liberty-reserve-one-world-s-largest.
54. Moser, M., R. Böhme, and D. Breuker. An inquiry into money laundering tools in the Bitcoin ecosystem. in eCrime Researchers Summit (eCRS), 2013. 2013. IEEE.
55. Greenberg, A. Dark Wallet Aims To Be The Anarchist's Bitcoin App Of Choice. 2013 [cited 2015 September 29, 2015]; Available from: http://www.forbes.com/sites/andygreenberg/2013/10/31/darkwallet-aims-to-be-the-anarchists-bitcoin-app-of-choice/.
56. Buterin, V. Why The Bitcoin Greenlist is Structurally Dangerous to the Bitcoin Ecosystem. 2013 [cited 2015 Septem-

ber 28, 2015]; Available from: https://bitcoinmagazine.com/articles/why-the-bitcoin-greenlist-is-structurally-dangerous-to-the-bitcoin-ecosystem-1384492133.

57. Hill, K. Sanitizing Bitcoin: This Company Wants To Track 'Clean' Bitcoin Accounts. 2013 [cited 2015 September 29, 2015]; Available from: http://www.forbes.com/sites/kashmirhill/2013/11/13/sanitizing-bitcoin-coin-validation/.
58. Cohen, B. Obama Initiative Spawns Identity Based Bitcoin Greenlist. 2013 [cited 2015 September 28, 2015]; Available from: https://bitcoinmagazine.com/articles/obama-initiative-spawns-identity-based-bitcoin-greenlist-1384277984.
59. Back, A. Coin Validation misunderstands fungibility and could destroy bitcoin. 2013 [cited 2015 September 29, 2015]; Available from: https://bitcointalk.org/index.php?topic=333882.0.
60. djsjjd. Sanitizing Bitcoin: This Company Wants To Track 'Clean' Bitcoin Accounts. 2014 [cited 2015 September 29, 2015]; Available from: https://www.reddit.com/r/Bitcoin/comments/1qj7sw/sanitizing_bitcoin_this_company_wants_to_track/cddie7r.
61. Piuk. What is taint? 2012 [cited 2015 September 19, 2015]; Available from: https://bitcointalk.org/index.php?topic=92416.msg1018943#msg1018943.
62. Wikipedia. Tor (anonymity network). 2015 September 19, 2015 [cited 2015 September 29, 2015f]; Available from: https://en.wikipedia.org/wiki/Tor_(anonymity_network).
63. Dingledine, R., N. Mathewson, and P. Syverson, Tor: The second-generation onion router. 2004, DTIC Document.
64. Bitcoin Wiki. Tor. 2011 September 3, 2015 [cited 2015 September 29, 2015].
65. Biryukov, A., D. Khovratovich, and I. Pustogarov. Deanonymisation of clients in Bitcoin P2P network. in 2014 ACM SIGSAC Conference on Computer and Communications Security. 2014. ACM.
66. Kaminsky, D. Black Ops of TCP/IP. 2011 [cited 2015 September 29, 2015]; Available from: http://dankaminsky.com/2011/08/05/bo2k11/.
67. Maxwell, G. CoinJoin: bitcoin privacy for the real world. 2013 [cited 2015 September 12, 2015]; Available from: https://bitcointalk.org/index.php?topic=279249.0.

68. Maxwell, G., I taint rich. 2013.
69. Martin, P. and A. Taaki. Anonymous Bitcoin Transactions. 2013 [cited 2015 August 25, 2015]; Available from: https://sx.dyne.org/anontx/.
70. Maxwell, G. CoinSwap: Transaction graph disjoint trustless trading. 2013 [cited 2015 September 12, 2015]; Available from: https://bitcointalk.org/index.php?topic=321228.0.
71. Bonneau, J., et al., Mixcoin: Anonymity for Bitcoin with accountable mixes, in Financial Cryptography and Data Security. 2014, Springer. p. 486-504.
72. Hearn, M. Merge avoidance A note on privacy-enhancing techniques in the Bitcoin protocol. 2013 [cited 2015 September 28, 2015]; Available from: https://medium.com/@octskyward/merge-avoidance-7f95a386692f.
73. Coutu, O. Decentralized Mixers for Bitcoin. 2013 [cited 2015 September 30, 2015]; Available from: https://www.youtube.com/watch?v=6hc8qaR_Fok.
74. Wikipedia. Clos Network. 2008 5 March 2016 [cited 2016 6 March]; Available from: https://en.wikipedia.org/wiki/Clos_network.
75. Androulaki, E., et al., Evaluating user privacy in bitcoin, in Financial Cryptography and Data Security. 2013, Springer. p. 34-51.
76. Reid, F. and M. Harrigan, An analysis of anonymity in the bitcoin system. 2013: Springer.
77. Meiklejohn, S., et al., A Fistful of Bitcoins: Characterizing Payments Among Men with No Names. USENIX ;login:, 2013.
78. Voskuil, E. About Bitcoin Explorer. 2014 15 February 2015 [cited 2016 9 January 2016]; Available from: https://github.com/libbitcoin/libbitcoin-explorer/wiki.

TENTANG PENULIS

DIMAZ ANKAA WIJAYA

Penulis menyelesaikan pendidikan Sarjana Komputer di Universitas Gadjah mada pada tahun 2007, kemudian memulai karir sebagai pengembang perangkat lunak. Dimaz lalu melanjutkan studi melalui beasiswa LPDP dalam program Master of Networks and Security di Monash University dan kini sedang melakukan penelitian tentang Bitcoin dan *cryptocurrency*. Dimaz dapat dihubungi melalui e-mail: dimaz@kriptologi.com.

Lampiran A: Bitcoin OpCode

Lampiran ini merupakan penjelasan ringkas tentang berbagai jenis OpCode Bitcoin yang dapat digunakan untuk menyusun script [30].

Konstanta

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_0, OP_FALSE	0	0x00	Nothing	Nilai kosong	Nilai kosong dimasukkan ke dalam *stack*.
	1-75	0x01 – 0x4b	Spesial	Data	Menunjukkan jumlah *byte* data berikutnya yang akan dimasukkan ke dalam *stack*.
OP_PUSHDATA1	76	0x4c	Spesial	Data	Nilai *byte* berikutnya menunjukkan jumlah *byte* data berikutnya yang akan dimasukkan ke dalam *stack*.
OP_PUSHDATA2	77	0x4d	Spesial	Data	Nilai 2 *byte* berikutnya menunjukkan jumlah *byte* data berikutnya yang akan dimasukkan ke dalam *stack*.
OP_PUSHDATA4	78	0x4e	Spesial	Data	Nilai 4 byte berikutnya menunjukkan jumlah *byte* data berikutnya yang akan dimasukkan ke dalam *stack*.
OP_1NEGATE	79	0x4f	Nothing	-1	Nilai -1 dimasukkan ke dalam *stack*.
OP_1, OP_TRUE	81	0x51	Nothing	1	Nilai 1 dimasukkan ke dalam *stack*.
OP_2 – OP_16	82-96	0x52 – 0x60	Nothing	2-16	Nilai yang disebutkan dimasukkan ke dalam *stack*.

Kontrol program

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_NOP	97	0x61			Tidak melakukan apapun.
OP_IF	99	0x63	<ekspresi> if [blok 1] else [blok 2] endif		Jika nilai teratas dalam *stack* tidak bernilai 0, maka perintah dalam blok 1 dieksekusi. Setelah itu nilai teratas dalam *stack* dihapus.
OP_NOTIF	100	0x64			Jika nilai teratas dalam *stack* bernilai 0, maka perintah dalam blok 1 dieksekusi. Setelah itu nilai teratas dalam *stack* dihapus.
OP_ELSE	103	0x67			Jika perintah OP_IF atau OP_NOTIF atau OP_ELSE sebelumnya tidak dieksekusi, maka perintah ini dieksekusi. Demikian sebaliknya.
OP_ENDIF	104	0x68			Perintah untuk mengakhiri blok if-else. Setiap perintah OP_IF harus diakhiri dengan OP_ENDIF. Jika tidak, maka transaksi dianggap invalid. Script dengan OP_ENDIF tanpa didahului OP_IF juga dianggap invalid.
OP_VERIFY	105	0x69	True/False	Nothing/Fail	Jika nilai teratas dalam stack tidak bernilai TRUE, maka transaksi invalid.
OP_RETURN	106	0x6a	Nothing	Fail	Digunakan untuk memasukkan informasi tambahan ke dalam transaksi. Operasi OP_RETURN diikuti dengan 1 perintah pushdata untuk memasukkan data ke dalam stack. Sebuah transaksi tidak dapat memiliki lebih dari 1 transaksi OP_RETURN.

Manipulasi Stack

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_IFDUP	115	0x73	x	x x	Jika nilai paling atas dari stack tidak 0, maka duplikasi.
OP_DEPTH	116	0x74	Nothing	Ukuran stack	Menaruh jumlah nilai yang ada di dalam stack ke dalam stack
OP_DROP	117	0x75	x	Nothing	Menghapus nilai teratas stack.
OP_DUP	118	0x76	x	x x	Menduplikasi nilai teratas stack.
OP_NIP	119	0x77	x1 x2	x2	Menghapus nilai kedua teratas dari stack.
OP_OVER	120	0x78	x1 x2	x1 x2 x1	Menyalin nilai kedua teratas ke puncak stack.
OP_PICK	121	0x79	xn…x2 x1	xn.. x2 x1 xn	Menyalin nilai ke-n dari stack ke puncak stack.
OP_ROLL	122	0x7a	xn…x2 x1	… x2 x1 xn	Memindahkan nilai ke-n dari stack ke puncak stack.
OP_ROT	123	0x7b	x1 x2 x3	x2 x3 x1	Tiga nilai teratas dari stack diputar ke kiri sebanyak satu kali.
OP_SWAP	124	0x7c	x1 x2	x2 x1	Dua nilai teratas dari stack ditukar posisinya satu sama lain.
OP_TUCK	125	0x7d	x1 x2	x2 x1 x2	Nilai teratas dari stack disalin ke urutan ketiga.
OP_2DROP	109	0x6d	x1 x2		Menghapus 2 nilai teratas dari stack.
OP_2DUP	110	0x6e	x1 x2	x1 x2 x21 x2	Menyalin 2 nilai teratas dari stack.
OP_3DUP	111	0x6f	x1 x2 x3	x1 x2 x3 x1 x2 x3	Menyalin 3 nilai teratas dari stack.
OP_2OVER	112	0x70	x1 x2 x3 x4	x1 x2 x3 x4 x1 x2	Menyalin nilai ke-4 dan ke-3 ke puncak stack.
OP_2ROT	113	0x71	x1 x2 x3 x4 x5 x6	x3 x4 x5 x6 x1 x2	Memindah nilai ke-5 dan ke-6 ke puncak stack.
OP_2SWAP	114	0x72	x1 x2 x3	x3 x4 x1 x2	Menukar posisi 2 pasang nilai dari stack.

Operasi Level Bit

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_EQUAL	135	0x87	x1 x2	True/False	Menghasilkan 1 jika kedua input sama, jika tidak maka 0.
OP_EQUALVERIFY	136	0x88	x1 x2	Nothing/fail	Sama dengan OP_EQUAL tetapi dilanjutkan dengan operasi OP_VERIFY

Aritmatika

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_1ADD	139	0x8b	in	out	1 ditambahkan ke input.
OP_1SUB	140	0x8c	in	out	1 dikurangkan dari input.
OP_NEGATE	143	0x8f	in	out	Nilai negatif menjadi positif, demikian sebaliknya.
OP_ABS	144	0x90	in	out	Nilai diubah menjadi positif
OP_NOT	145	0x91	in	out	Jika input bernilai 0 atau 1, maka diubah. Jika tidak maka output bernilai 0.
OP_0NOTEQUAL	146	0x92	in	out	Menghasilkan 0 jika input bernilai 0, selain itu akan menghasilkan 1.
OP_ADD	147	0x93	a b	out	Nilai a ditambahkan ke b.
OP_SUB	148	0x94	a b	out	Nilai a dikurangi b.
OP_BOOLAND	154	0x9a	a b	out	Jika a dan b tidak bernilai 0, maka hasilnya 1. Selain itu hasilnya 0.
OP_BOOLOR	155	0x9b	a b	out	Jika a atau b tidak bernilai 0, maka hasilnya 1. Selain itu hasilnya 0.
OP_NUMEQUAL	156	0x9c	a b	out	Jika nilainya sama maka menghasilkan 1. Selain itu hasilnya 0.
OP_NUMEQUALVERIFY	157	0x9d	a b	Nothing/fail	Sama seperti OP_NOMEQUAL tetapi dilanjutkan dengan operasi OP_VERIFY
OP_NUMNOTEQUAL	158	0x9e	a b	out	Jika angka tidak sama maka hasilnya 1. Selain itu hasilnya 0.
OP_LESSTHAN	159	0x9f	a b	out	Jika a kurang dari b maka hasilnya 1. Selain itu hasilnya 0.
OP_GREATERTHAN	160	0xa0	a b	out	Jika a lebih besar dari b maka hasilnya 1. Selain itu hasilnya 0.
OP_LESSTHANOREQUAL	161	0xa1	a b	out	Jika a kurang dari atau sama dengan b, maka hasilnya 1. Selain itu hasilnya 0.
OP_GREATERTHANOREQUAL	162	0xa2	a b	out	Jika a lebih besar atau sama dengan b, maka hasilnya 1. Selain itu hasilnya 0.
OP_MIN	163	0xa3	a b	out	Hasilnya adalah nilai terendah antara a atau b.
OP_MAX	164	0xa4	a b	out	Hasilnya adalah nilai tertinggi antara a atau b.
OP_WITHIN	165	0xa5	x min max	out	Menghasilkan 1 jika nilai x berada di antara nilai min dan nilai max. Selain itu hasilnya 0.

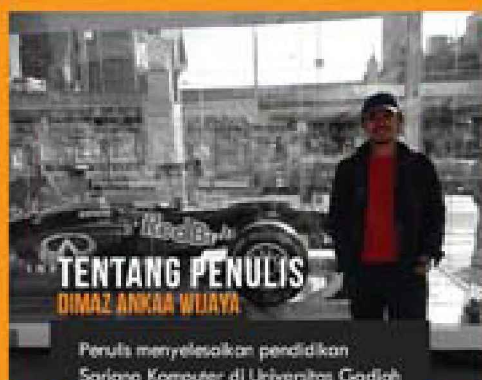
Kriptografi

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_RIPEMD160	166	0xa6	in	hash	Menghasilkan nilai hash menggunakan RIPEMD-160.
OP_SHA1	167	0xa7	in	hash	Menghasilkan nilai hash menggunakan SHA-1.
OP_SHA256	168	0xa8	in	hash	Menghasilkan nilai hash menggunakan SHA-256.
OP_HASH160	169	0xa9	in	hash	Menghasilkan nilai hash mengunakan SHA-256 dilanjutkan dengan RIPEMD-160.
OP_HASH256	170	0xaa	in	hash	Menghasilkan nilai hash menggunakan SHA-256 sebanyak 2 kali.
OP_CODESEPARATOR	171	0xab	Nothing	Nothing	Untuk memisahkan data yang digunakan untuk mengecek *digital signature*.
OP_CHECKSIG	172	0xac	sig pubkey	True/False	Seluruh output, input, dan skrip di-hash kemudian dilakukan pengecekan digital signature dan public key. Jika valid maka hasilnya 1, selain itu hasilnya 0.
OP_CHECKSIGVERIFY	173	0xad	sig pubkey	Nothing/fail	Sama seperti OP_CHECKSIG, tetapi dilanjutkan dengan OP_VERIFY.
OP_CHECKMULTISIG	174	0xae	x sig1 sig2… y pub1 pub2… y	True/false	X adalah jumlah minimum digital signature, sedangkan Y adalah jumlah total digital signature. Operasi ini membandingkan digital signature dengan public key. Digital signature dan public key harus diletakkan dengan urutan yang benar. Jika semua digital signature valid, hasilnya 1. Selain itu hasilnya 0. Berhubung terdapat bug, maka OP_0 harus ditambahkan yang nantinya akan dibuang dari stack.
OP_CHECKMULTISIGVERIFY	175	0xaf	x sig1 sig2… y pub1 pub2… y	Nothing/fail	Sama seperti OP_CHECKMULTISIG tetapi dilanjutkan dengan OP_VERIFY

LockTime

Kode	Opcode	Heksadesimal	Input	Output	Deskripsi
OP_CHECKLOCKTIMEVERIFY (sebelumnya OP_NOP2)	177	0Xb1	Nothing	Nothing/fail	Jika nilai teratas dari stack lebih besar daripada Locktime, maka transaksi invalid. Selain itu, transaksi invalid jika: 1. stack kosong. 2. nilai teratas stack negatif. 3. nilai teratas stack lebih besar dari 500 juta sementara LockTime kurang dari 500 juta atau sebaliknya. 4. sequence number sama dengan 0xffffffff. Deskripsi lengkap ada dalam dokumen BIP0065 [40].

Bitcoin merupakan sistem pembayaran digital yang diperkenalkan oleh Satoshi Nakamoto pada tahun 2008. Sistem ini merupakan terobosan baru yang memungkinkan orang untuk melakukan transaksi satu sama lain tanpa melalui trusted party (pihak ketiga yang dipercaya seperti bank). Menghapus trusted party di dalam sebuah sistem pembayaran mengharuskan verifikasi atas validitas transaksi keuangan harus dilakukan dengan cara yang berbeda, dan di sinilah peran kriptografi. Karena Bitcoin tidak membutuhkan *trusted party*, maka sistem ini dapat berjalan dalam sistem peer-to-peer di mana tidak ada satupun yang bertindak sebagai dedicated server, melainkan setiap komputer saling mengirimkan informasi terbaru, sehingga pada akhirnya semua komputer memiliki informasi yang sama. Bitcoin juga berjalan dalam sistem yang terdesentralisasi dengan tidak ada satu pihak pun yang menjadi pusat pengendali sistem.



Penulis menyelesaikan pendidikan Sarjana Komputer di Universitas Gadjah mada pada tahun 2007, kemudian memulai karir sebagai pengembang perangkat lunak. Dimaz lalu melanjutkan studi melalui beasiswa LPDP dalam program Master of Networks and Security di Monash University dan kini sedang melakukan penelitian tentang Bitcoin dan cryptocurrency.

Dimaz dapat dihubungi melalui e-mail: dimaz@kriptologi.com.





